

To the Graduate Council:

I am submitting herewith a thesis written by Peyman H. Dehkordi entitled "A Microprocessor-based Video-scanning Trash-meter." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Robert W. Rochelle

Robert W. Rochelle, Major Professor

We have read this thesis
and recommend its acceptance:

Donald W. Bouldin

Robert E. Bachmeyer

Accepted for the Council:

Lowminkel

Vice Provost
and Dean of The Graduate School

A MICROPROCESSOR-BASED VIDEO-SCANNING
TRASH-METER

A Thesis
Presented for the
Master of science
Degree
The University of Tennessee

Peyman Hosseini Dehkordi

June 1985

ACKNOWLEDGEMENTS

I would like to acknowledge and pay special thanks to a few important people that made my higher education possible. First, and most important, this would not have been possible without my parents criticism and encouragement. They have been behind me in every effort to achieve my life-long goal.

I owe a great deal of appreciation to Mr. Hossein Ghorashi for his patience and technical advisement. I would also like to thank Professors J.R. Roth and R.W. Rochelle for granting me the research-assistantship which made my education financially possible. Last but not least, I would like to acknowledge appreciation to professors R.E. Bodenheimer and D.W. Bouldin for serving on my committee.

ABSTRACT

This thesis report describes the hardware, software design and testing of a complete microprocessor-based image-analysis system called a "trash-meter." The trash-meter is a single STD-card Z-80A based system which performs a two-dimensional cotton surface scanning using a high-resolution charge-coupled-device camera. The trash-meter measures the amount of the foreign content of cotton referred to as "trash" in terms of trash area and number of distinct trash spots present on the cotton surface. Once a picture frame is digitized and stored (in real time) and a complete analysis is done by the system software, the results (trash area and trash count) are sent to a host computer serially.

TABLE OF CONTENTS

CHAPTER	PAGE
INTRODUCTION	1
I. HARDWARE DESIGN	5
Central Processing Unit	
Programmable Usart	
The System Memory	
The Video Camera And Video Signal	
The Direct Memory Access (DMA) controller	
Miscellaneous	
II. SOFTWARE DESIGN	48
Algorithm	
Software Considerations	
III. PUSH TO TEST AND TROUBLE SHOOTING	77
IV. THE TRASH-METER EVALUATION	81
Particle-Size Sensitivity	
Area linearity and Stability	
Summary And Conclusion	
LIST OF REFERENCES	90
APPENDIXES	92
A. Program Listing of The System	
B. The Format of The Digitized Picture in The RAM	
C. The Complete Schematic of The Trash-meter Hardware	
VITA	133

LIST of FIGURES

FIGURE	PAGE
1. The General Block Diagram of The Trash-meter	3
2. The Z-80A Pin diagram	6
3. The CPU Block	9
4. The 8251A USART	12
5. The Block Diagram of The 8251A Interface	14
6. The 8251A Interface to The System Bus	15
7. The Block Diagram of The Memory section	19
8. The Memory Interface To The System Bus	20
9. The Memory Chips Pin Diagram	21
10. The System Memory Map	23
11. The Fairchild CCD-3000 Camera	25
12. The Interlaced Video Format	26
13. The RS-170 Video Format	28
14. The CCD-3000 Timing Diagram	29
15. The Block Diagram of The DMA Controller	31
16. The Circuit Diagram of The DMA Controller Part1	32
17. The Circuit Diagram of The DMA Controller Part2	33
18. The Circuit Diagram of The DMA Controller Part3	34
19. The Circuit Diagram of The DMA Controller Part4	35
20. The Timing Diagram 1	38
21. The Timing Diagram 2	40
22. The Timing Diagram 3	42
23. The Timing Diagram 4	44
24. The Format of The Digitized Video In The RAMS	46

25. Trash Area And Trash Count of Cotton	49
26. Partitioning Format of The Cotton Image	50
27. The Trash Spots And Scan Lines Part1	51
28. The Trash Spots And Scan Lines Part2	54
29. The Camera Observation Window	57
30. The Format of One Scan Line in The RAM	58
31. The Format of The Table2	60
32. The Format of The Table1	62
33. The Serial Communication Format	63
34. The Flow Chart of The Power-up Routine	65
35. The General Flow Chart of The Interrupt Routine	67
36. The Flow Chart of The First Part of Interrupt Routine	68
37. The Flow Chart of The Second Part of The Interrupt Routine	73
38. The Flow Chart of The Push-to-test routine	78
39. The Test Setup	82
40. The Particle-sensitivity Graph	84
41. The Calibration Papers	85
42. The Area-Linearity Graph	88
43. The Complete Schematic of The Trash-meter Hardware	131
44. The Trash-meter Circuit Board	132

INTRODUCTION

"Trash" is a common term used in the cotton industry for the foreign content of cotton . Not very many years ago, cotton was cleanly picked from the plant by hand, consequently cotton gins produced bales with a low trash content. Today's production techniques, however, have drastically increased the amount of foreign matter in harvested cotton. Mills, the cotton consumers, must cope with the problem of variable and increasing trash fed into their sophisticated and sensitive production equipment which demands super-clean cotton for efficient operation. The mill therefore is caught between dealing with a raw product of variable trash content and production equipment that responds most efficiently to consistency. The mill's opening, cleaning, and carding processes can not solve the problem of variation, so the mill must gain control over variable trash content through purchasing, mix selection, and laydown. Consequently, the need for some kind of trash-testing equipment for cotton is quite pressing. Once the visible trash content of each bale is known, purchasing specifications can be maintained, mix selection made, and laydown patterns designed.

Until now, the mills have had two sources for estimation of trash content: the human eye and the mechanical separator. The human eye brings with it a limited attention span and a judgment drift of about 12 percent. The mechanical separator uses a very slow mechanical process, making it economically unfeasible to test every bale. The Spinlab 935 Trashmeter is one of the first instruments capable of testing the visible content of every bale with speed and accuracy.

The 935 trash-meter contains a single STD-card built into a Spinlab 900 high-volume testing instrument. It is a microprocessor based instrument, which performs a two-dimensional surface scan using a high-resolution CCD camera. This camera employs a monolithic 380 by 488 element charge-couple device as its image sensor. A field of view containing 350 by 450 pixels is used to analyze the cotton through a 3.5" by 3.5" window. Each pixel represents either a white or black spot on the cotton (i.e. either cotton or trash). By counting the number of trash pixels present on the surface of the cotton, the trashmeter computes the trash area, and number of distinct trash spots.

The general block diagram of the trash meter is shown in Figure 1. The basic principle of the operation is as follows:

1. A sample of cotton is placed in front of the camera, and the "start" signal is activated.
2. Upon activation, a complete picture frame of the cotton will be digitized and stored in the RAMs through the custom made direct-memory-access (DMA).
3. Next, the Z-80A microprocessor will take over the bus and analyze the data.
4. After computing the area, count, and size of the trash, the Z-80A microprocessor then transmits the results to a host computer through a programmable universal synchronus/asynchronus receiver/transmitter (USART).
5. Once successful communication with the host computer has been made, the process is repeated.

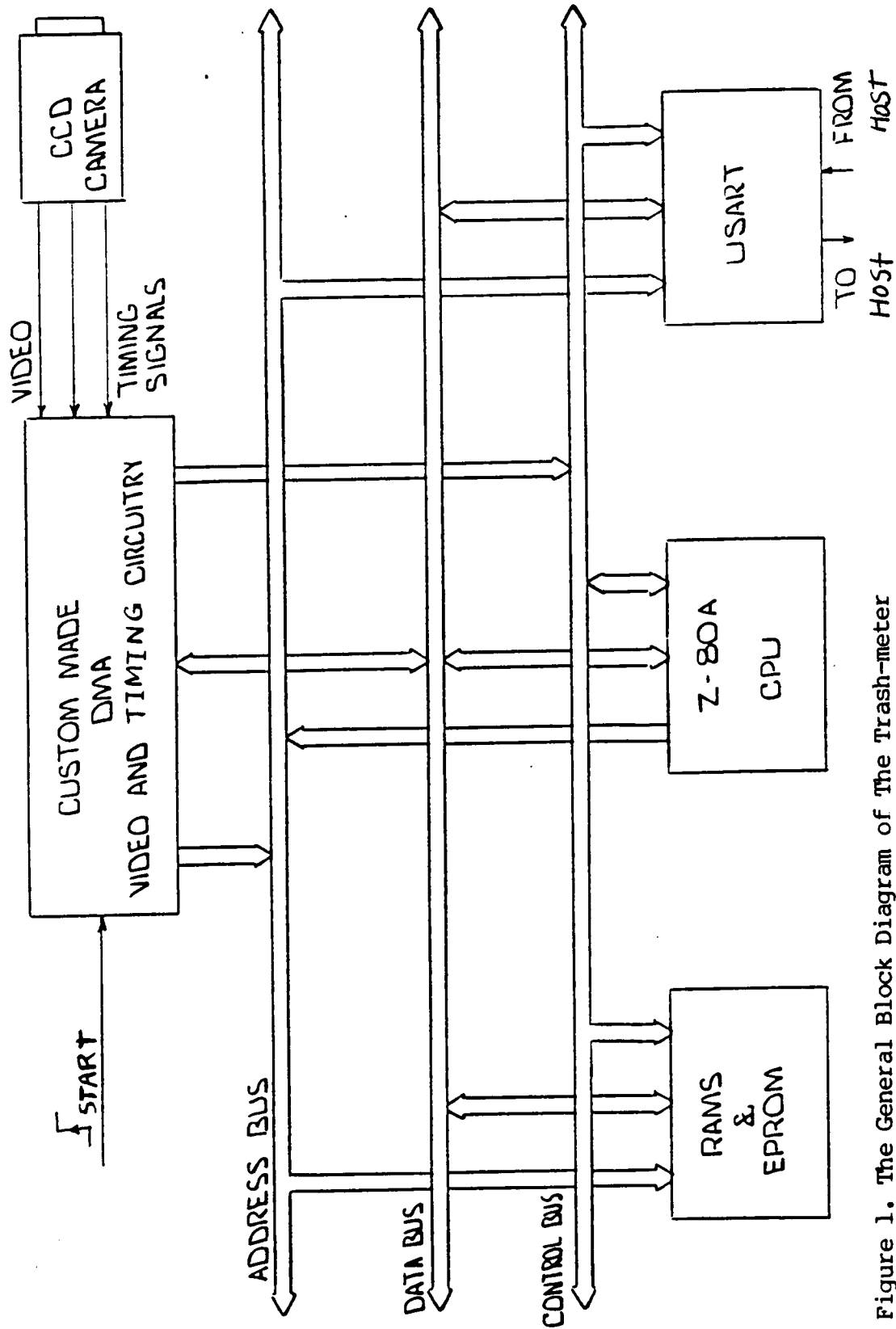


Figure 1. The General Block Diagram of The Trash-meter

This thesis will be organized in four chapters. Chapter I will detail the design philosophy of the hardware as well as all of the necessary timing diagrams. Chapter II will discuss the software design philosophy including the algorithms and the flow charts used. Chapter III describes some other features of this instrument such as the push-to-test capability and hardware trouble-shooting procedure. The last chapter evaluates the instrument and describes its capabilities and limitations.

CHAPTER I

HARDWARE DESIGN

The general block diagram of the overall system is shown in Figure 1. There are mainly four blocks: 1) the CPU block 2) the memory block 3) the custom made DMA and 4) the serial I/O . The following paragraphs will describe the function of each block in detail and explain the design philosophy as well as the timing diagrams involved.

A. Central Processing Unit

The heart of the trash-meter is a Z-80A CPU. This particular processor was chosen mainly because of its powerful instruction set and the availability of its supporting peripherals. Another major reason was the availability of the HP-64000 software and hardware development system for the Z-80A.

The CPU communicates with the other blocks shown in Figure 1 through tri-state data, address, and control buses. Unfortunately, the CPU communicates with the bus through no buffer because of lack of room on the board. However; care has been taken to make sure that the CPU is not overloaded.

The Z80-A CPU chip is a 40-pin dual in-line package shown in Figure 2. The pins are logically grouped according to their function rather than the actual physical representation.

The address bus is represented by A0-A15 which are tri-state output. Up to 64K of memory may be addressed through these address lines. D0 through D7 represent a bidirectional tri-state bus.

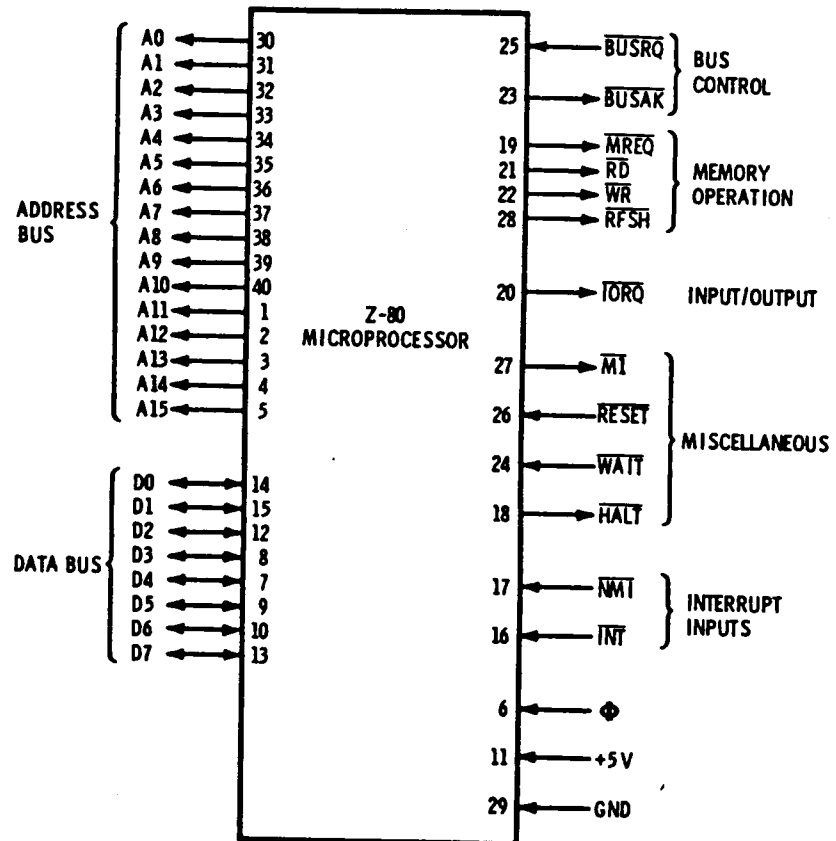


Figure 2. The Z-80A Pin diagram

The transfer of data is done through this bus. The third bus involved with the Z80-A is the control bus. The control bus will carry the control signals between the CPU and the external devices. There are four signals associated with memory operation. The MREQ, memory request, is a tri-state active-low output indicating when the address bus holds a valid address. The RD and WR signals are also tri-state active-low signals. The Z-80 pulls either RD or WR low when signal MREQ goes low to perform a read or write operation from or to memory. RFRSH signal is mainly used to refresh the dynamic memories. IORQ is a tri-state active-low output which goes low with either RD or WR to perform an input or output from an I/O device on the bus.

There are two bus control signals associated with the Z-80. The input signal BUSRQ is an active-low input which is generated by an external device to gain control over the bus like the DMA controller. The other signal is BUSAK which is an active-low output. It is pulled low by the CPU in response to BUSRQ to signify that the address, data, and control buses are now in tri-state and a DMA operation can take place.

The Z-80 CPU supports two kinds of interrupts, non-maskable interrupts (NMI) and maskable interrupts (INT). The CPU can be interrupted at any time by bringing the NMI input momentarily low. The CPU will recognize the NMI by saving the current program counter (PC) and transfers the program control to memory 66H. The main interrupt request is the signal INT that is supplied by an external device. The CPU recognizes this INT at the end of the current instruction if the interrupt is enabled by the user program. The CPU

acknowledges the interrupt by bringing the IORQ and M1 lines low and services the interrupt according to the interrupt mode set by the user program. Signal M1 is an active-low output signal and indicates that the CPU is in the fetch cycle of an instruction. Slow memories and I/O devices communicate with the Z-80A by pulling the WAIT input low. As long as the WAIT signal is "low", the CPU will mark time while the slow memory or I/O device is responding to the CPU. RESET is an active-low input which resets the CPU. Upon receiving a reset pulse, the maskable interrupt will be disabled and all CPU tri-states buses go to the high-impedence state, and all output control signals go to the inactive state. HALT is an active-low output that goes low during the time that a HALT instruction is being executed. When the CPU is in the halt state, it performs a NOP (no operation) instruction to ensure proper memory refresh and recognition of the interrupts.

Figure 3 shows a more detailed picture of the CPU block. The WAIT input is pulled "high" because all memory and I/O devices communicate with the CPU without using any "wait" states. The Z-80A is clocked with a 4MHZ clock which comes to the board through the STD bus. The push-button reset signal also comes to the board through the STD bus and is run through two NAND gates for buffering and polarity purposes. The NMI line is usually pulled "high"; however, it may be brought "low" by switch1 for self-testing. The state of the HALT output is monitored by the led1. The led1 lights up any time the CPU is executing a "halt" instruction. The led1 and switch1 are used for self-testing purposes which are explained in chapter 3. The bus acknowledge (BUSAK) is also run through two inverters for buffering.

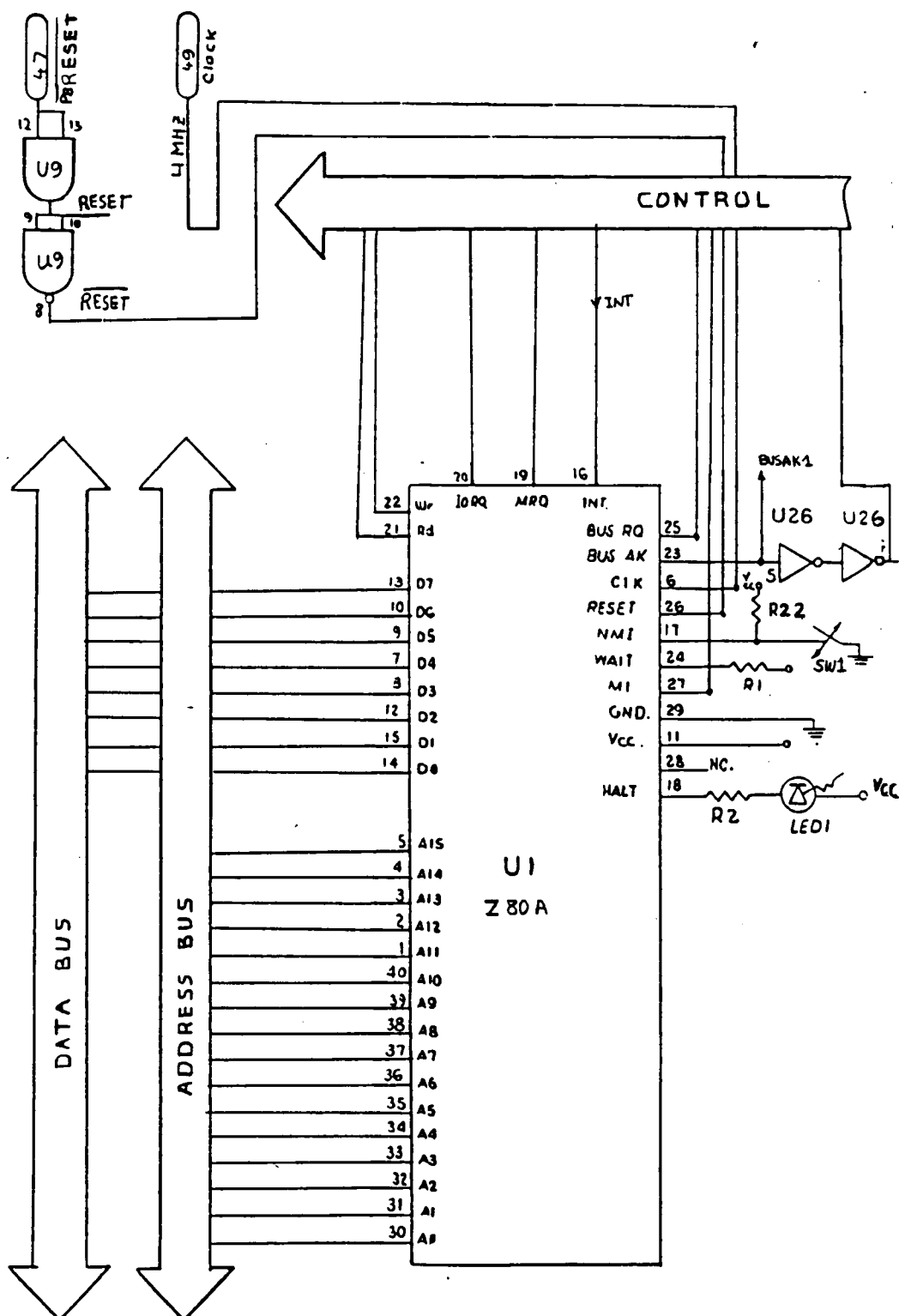


Figure 3. The CPU Block

B. Programmable Usart

Communication of a microprocessor-based system with the outside world is one of its most important features. In a serial communication environment an interface must convert parallel-format system data into serial format for transmission and convert incoming serial-format data into parallel system data for reception. The interface device must also delete or insert bits or characters that are functionally unique to the communication technique. The interface must also be easy to use, interface to the system bus without any extra hardware, and will be fully software programmed. In essence, the interface should appear "transparent" to the CPU, a simple input, output of byte oriented system data.

Considering all the above facts, the INTEL 8251A was chosen as the communication interface for the trash-meter. The INTEL 8251A is the enhanced version of the industry standard INTEL 8251 Universal Synchronus/Asynchronus Receiver/Transmitter (USART) designed for data communication with 8-bit microprocessors like the Z-80A. The 8251 is used as a peripheral device and is programed by the CPU to operate using virtually any serial data-transmission technique presently in use. The USART accepts data characters from the CPU in parallel format and then converts them into serial format. The USART will signal the CPU whenever it can accept new character for transmission or whenever it has received a character for the CPU. The CPU can read the complete status of the USART at any time. These include data

transmission errors and control signals such as Txempty, Rxrdy, and Rxrdy.

The functional block diagram of the 8251 is shown in Figure 4. It is capable of synchronus and asynchronous operations and has all the neccessary control signals for interfacing to a modem. The complete functional definition of the 8251A is programmed by the system's software. A set of control words must be sent out by the CPU to initilize the 8251A to support the desired communication format. These control words will program the Baud rate, Character length, number of stop bits, Synchronous/Asynchronous operation, Even/Odd/None parity, etc. Once programmed, the 8251A is ready to perform its communication functions. Because of the high degree of complexity involved with the 8251A, only the features used in the trash-meter will be described.

D0-D7 is a 3-state bidirectional buffer which interfaces the 8251A to the system data bus. Data is transmitted or received by the buffer upon execution of input (IN) or output (OUT) instructions from the CPU. The command status, data-in and data-out registers are separate 8-bit registers communicating with the system bus through the data bus buffer. A "high" on RESET input forces the 8251A into the idle mode and will remain in this mode until a new set of control words is written into the 8251. The CLK input is used to generate the internal device timing and is normally connected to the system clock. All the input and output operations to the 8251A is done by using the WR and RD inputs. A low on the WR input informs the 8251A that the CPU is writing data or control words to the 8251A. A "low" on RD

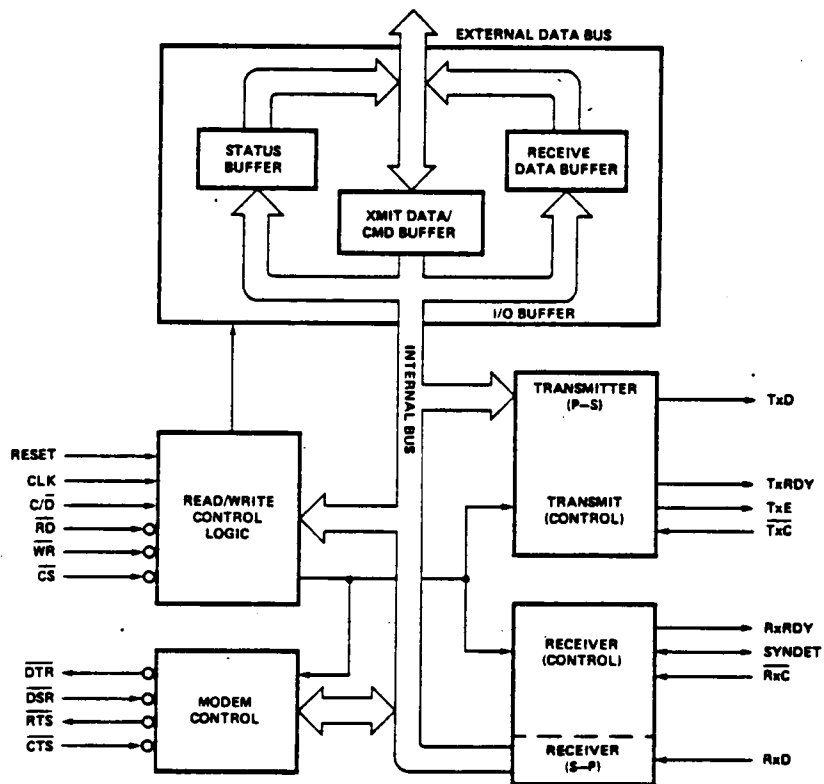


Figure 4. The 8251A USART

informs the 8251A that the CPU is reading data or status. C/D input is used in conjunction with WR and RD to inform the 8251A that the word or data bus is either a data character, control character or status information. No reading or writing will occur unless the device is selected by bringing the CS input "low." When CS is high the data bus is in the float state and RD and WR have no effect on the chip. The composite serial input and output data are received and transmitted through Rxd and Txd pins. Txc is the transmitter clock which controls the serial character transmission rate. In asynchronous mode the Txc frequency is a multiple of the actual baud rate. Two bits of the mode instruction select the multiple to be 1, 16, or 64 times the baud rate. The receiver clock (Rxc), controls the frequency of incoming data. The Rxc is a multiple of the actual baud rate like Txc.

The 8251A sits on the bus as an I/O device and communicates with the CPU through the Z-80A I/O instructions. Any time that the CPU wants to read or write from the 8251A, it places the I/O base address for the 8251A selection on the bottom half of the address bus (A0 through A7). At the same time the CPU brings the I/O request (IORQ) line low to initiate an I/O operation.

Figure 5 and 6 show how the 8251A interfaces with the rest of the system buses. 74LS520 is an 8-bit magnitude comparator which compares the M1 and A1 through A7 with the user preset 8251A I/O base address. M1 is always high during an I/O operation. Therefore using the M1 will make sure that the 8251A never gets selected unless during a proper I/O cycle. The G input of the

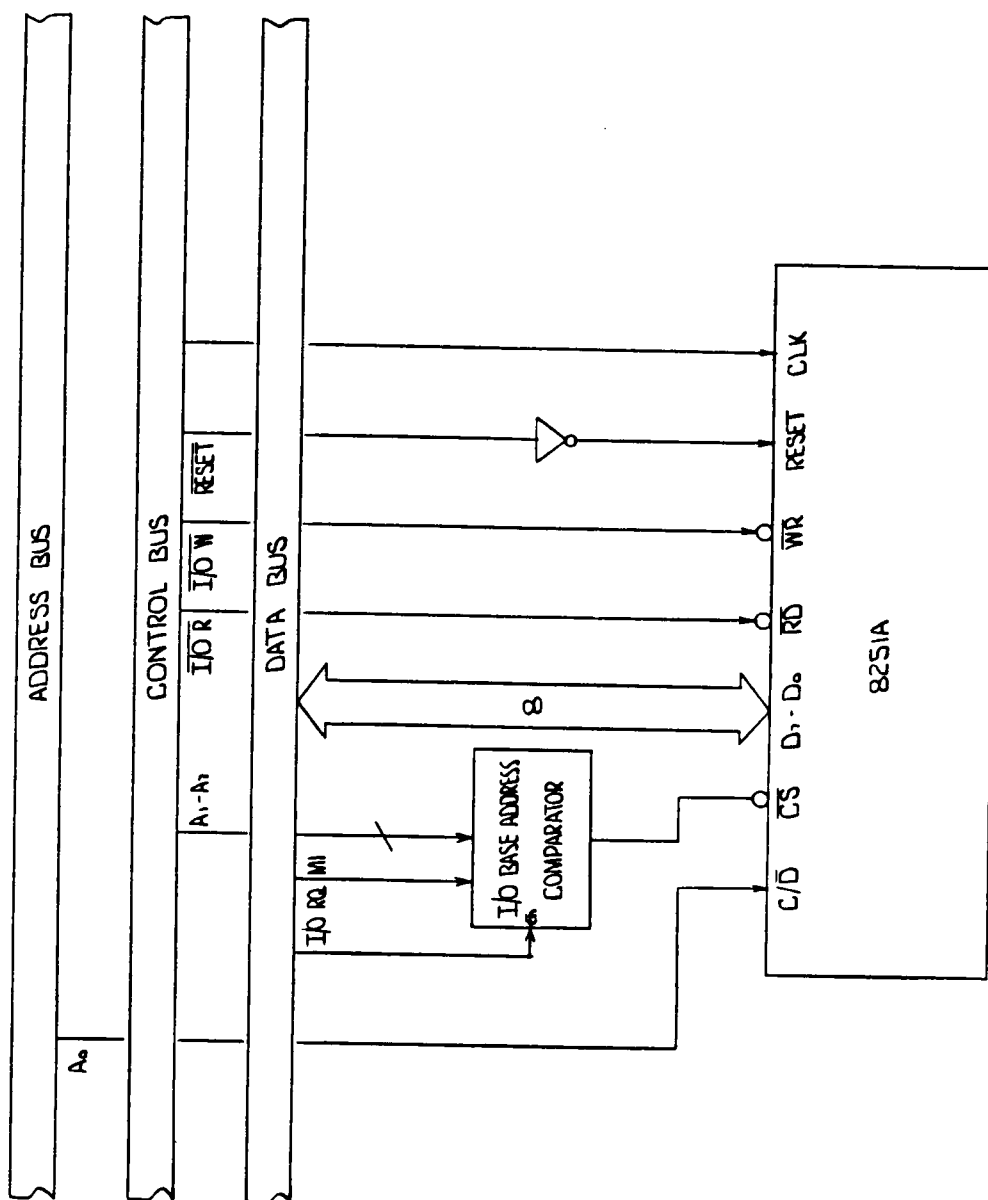


Figure 5. The Block Diagram of The 8251A Interface

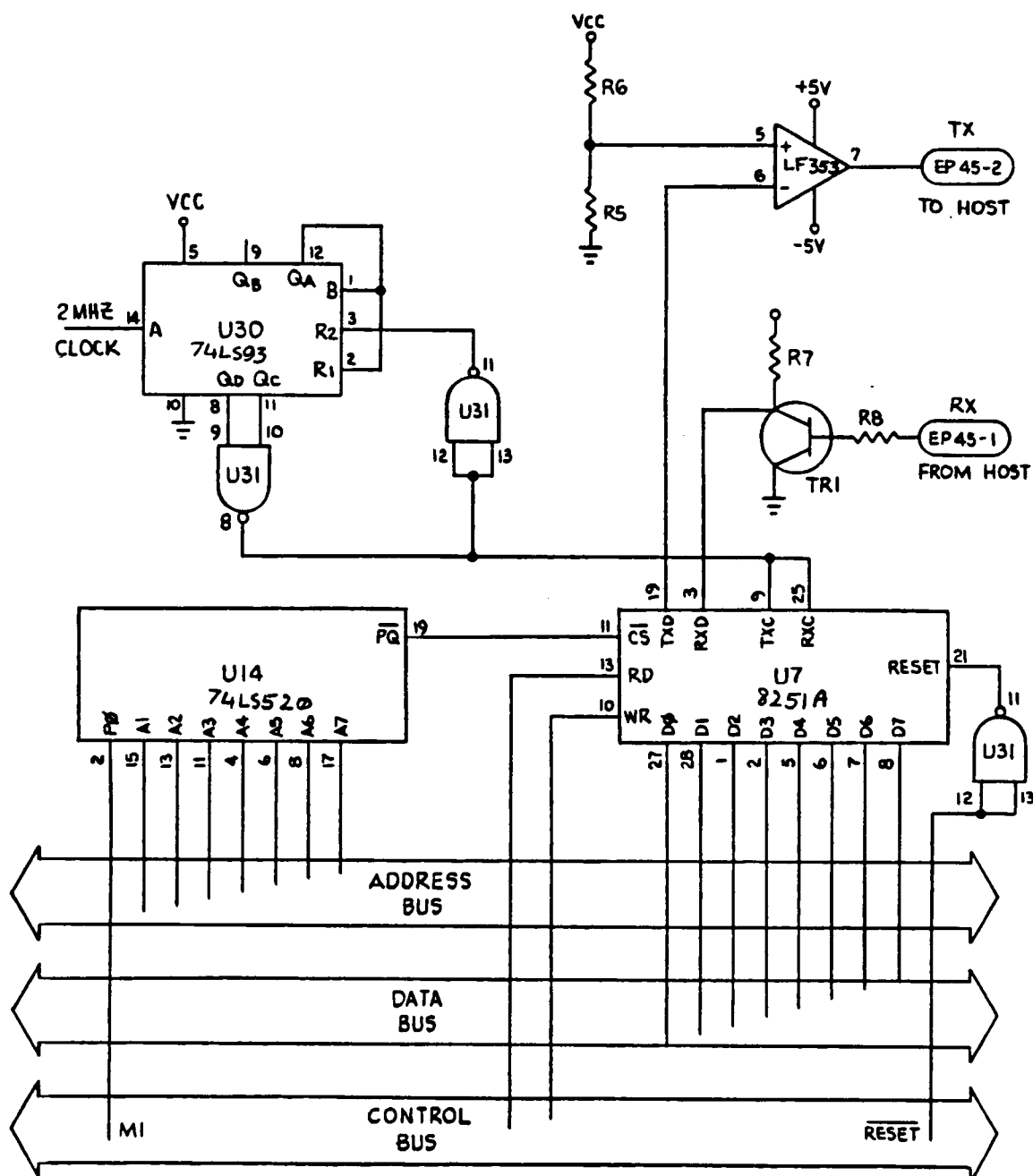


Figure 6. The 8251A Interface to The System Bus

of the comparator is connected to the IORQ line. This will allow the comparator to compare the M1 and A0-A7 with the I/O base address only when there is an I/O instruction in process. In case of a match, the output of the comparator will go low and enables the 8251A. The C/D input of the 8251A is connected to the address line A0. This will assign two consecutive I/O addresses to the 8251A control and data register. The Rd and Wr inputs are also connected to the system Rd and Wr for I/O read and write operation.

Since the maximum clock frequency for the 8251A is less than 4MHz, the 4MHz system clock is divided by two by using a D flip-flop to create a 2MHz clock. Unfortunately a user-selectable jumper for the baud-rate setting was not feasible because of the density of the board. 74LS93 is a 4-bit counter which is used to divide a 2MHz signal to 153.6KHz for the 8251A Rxc and Txc inputs. By proper programming, the actual transmission and receive baud rate could be 1, 1/16, 1/64 of the 153.6KHz (153,600 or 9,600 or 2,400 baud).

The communication to the host computer is assumed to be done through the RS-232C port. Therefore, a level change must be done to all incoming and transmitted data because the 8251A Rxd and Txd are TTL compatible. An incoming character level from a host computer ranges from -12V to +12V (i.e. logic "0" = +12V, logic "1" = -12V). The transistor TRL (Figure 6) is used as a switching transistor to convert the RS-232C logic level to TTL logic level. A logic "low" from the host (+12V) will turn the transistor on and causes the

collector is pulled to Vcc. Since -12V supply is not available on the board, the transmitted characters are only shifted to +5 and -5 volts. The operational amplifier shown in Figure 6 is used as a voltage comparator. The transmitted characters from the 8251A are compared to a constant 1.5V D.C. present on the + input of the op-amp. This will transform a "low" and "high" TTL logic level to +5 and -5 volts.

C. The System Memory

Another important part of the trash-meter system is the memory block. The memory block can be divided into two sub-blocks. The first sub-block is the Random Access Memory (RAM) which is a read and write memory. The other one is an Erasable Programmable Read-Only Memory (EPROM) which is a read-only-memory. There are a number of factors which were considered for choosing a memory product such as cost, power consumption, performance, memory architecture and organization, and size of the memory. Each of these factors plays an important role in the final selection process.

Generally, the term performance relates to how fast the device can operate in a given system environment. For example, in the case of the trash-meter, the RAMs have to be fast enough to communicate with the Z-80A and the DMA block without inserting any WAIT states to take advantages of the full speed of the Z-80A and the DMA. Another important term considered was the size and organization of the memory. By "size and organization" it is meant the number of bits and the number of bits per word in the memory chip. Since the trash-meter was originally designed to

be a single STD size card, a very high density RAM had to be used. Also, since the Z-80A is byte-oriented processor, the memory had to be eight-bits wide. Power consumption is another factor which was considered. Higher power consumption requires bigger power supplies, more cooling and reduced device density per board all affecting cost and reliability.

Figure 7 and 8 shows a more detailed picture of the memory block. Considering all the above, the Hitachi HM6264LP-15 was chosen to be used as the read-write memory. HM6264-LP15 is an 8K-by-8 high-speed static RAM with maximum access time of 150 ns which is compatible with the Z-80A and the DMA circuitry. Using static RAM also makes the design job much easier because the designer does not have to be concerned with memory refreshing. Its high density, single +5 volt supply, low power consumption, and directly TTL compatibility are some other features of the HM6264LP-15.

The EPROM shown in Figure 8 is an Intel 2764 which holds all the system software for the Z-80A CPU. It is an 8K-by-8 EPROM using a single +5-volt supply which gives lots of extra locations for feature software expansion or modifications. The standard 2764 access time is 250 ns which allows communication with the CPU without the addition of WAIT states. The pin diagrams of the HM6264LP-15 and Intel 2716 are shown in Figure 9. D0 through D7 are tri-state data lines to and from the memory. The address lines are A0-A12 which address one of the 8192 locations inside each memory chip. As long as the CS is "high", the memory chip will not respond to any READ or WRITE operation and stays in the "stand by" mode which

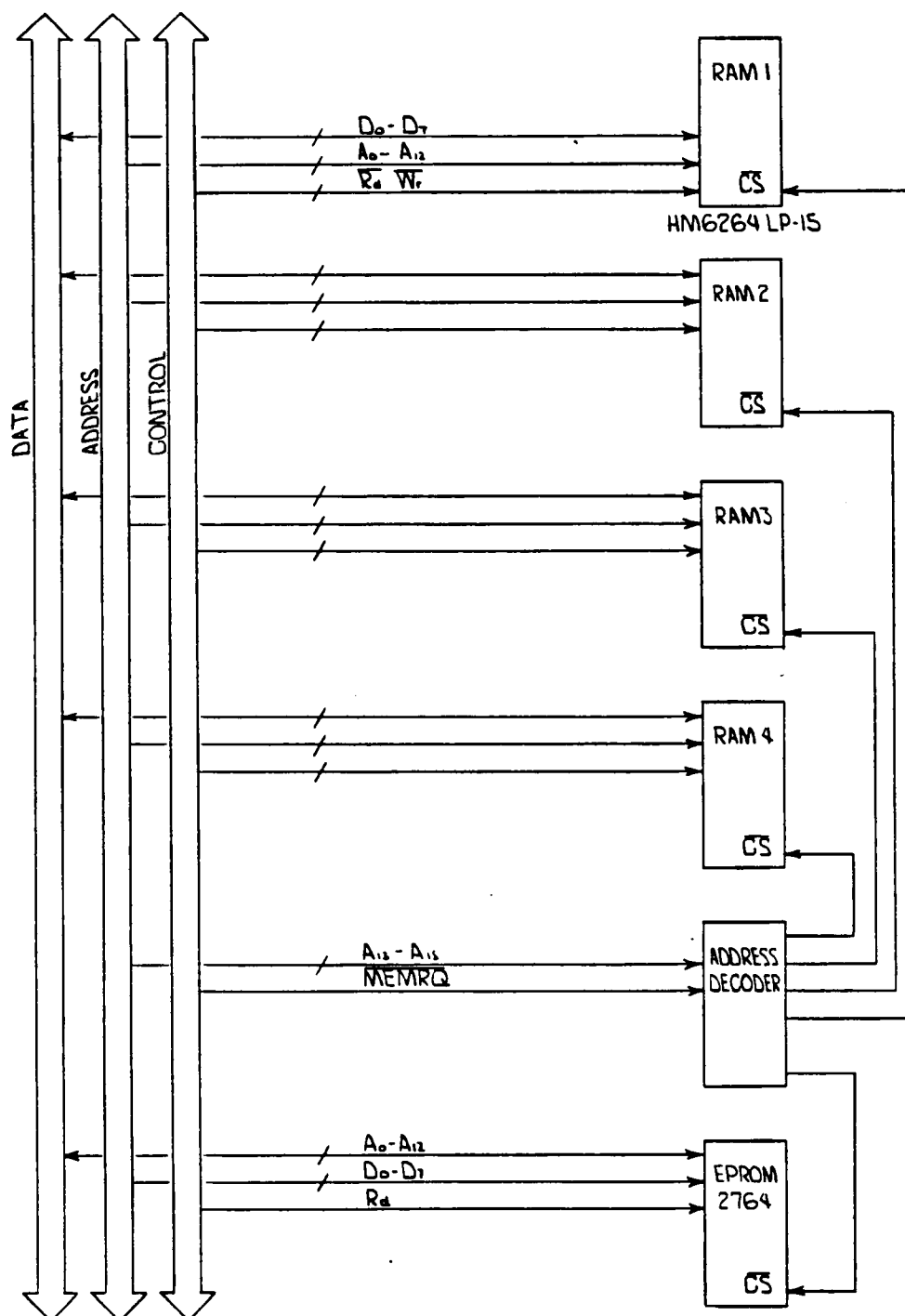


Figure 7. The Block Diagram of The Memory section

NOTE: U3, U4, U5 ARE IDENTICAL TO U2
WITH EXACT SAME CONNECTIONS EXCEPT
FOR PIN 20, WHICH IS SHOWN.

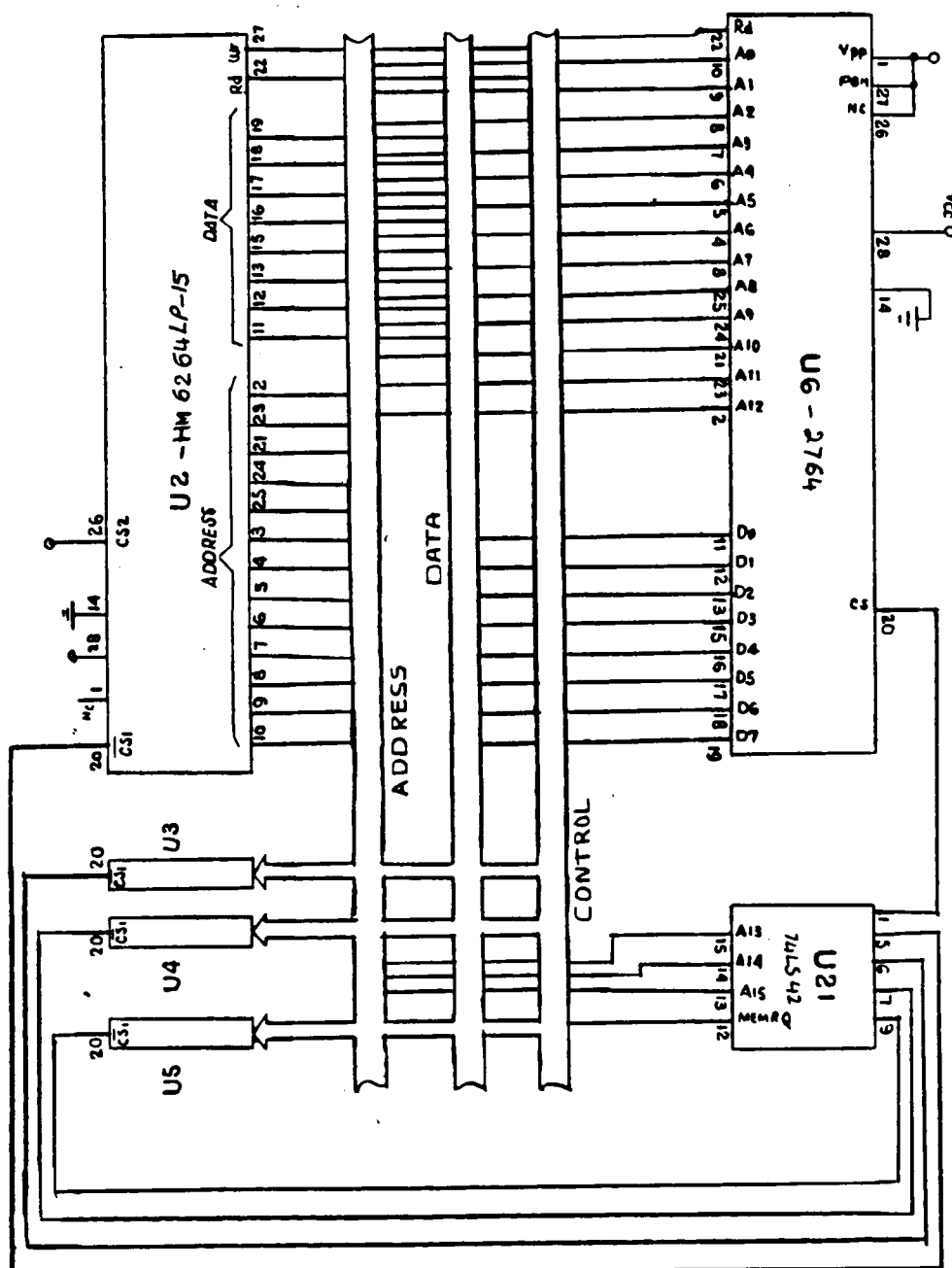
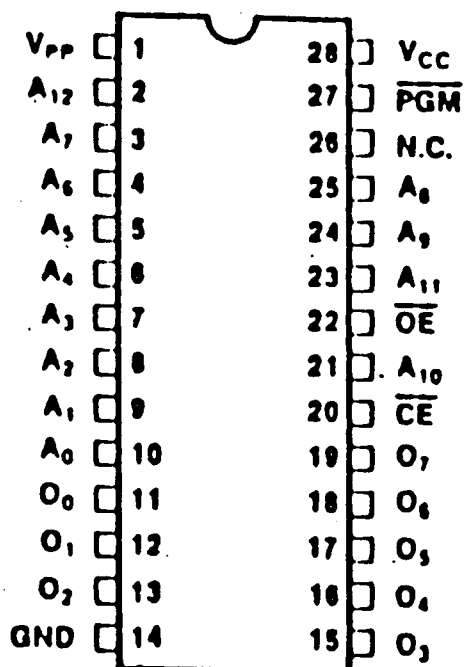
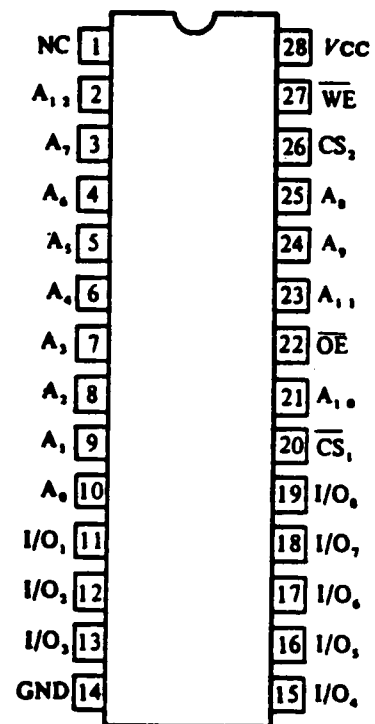


Figure 8. The Memory Interface To The System Bus



(A)



(B)

Figure 9. The Memory Chips Pin Diagram

has a very low power consumption. When CS goes "low", a READ or a WRITE operation may take place according to the state of the RD or WR pins. If RD (or OE in EPROM) input goes "low" while CS is low too, a READ operation will take place. A WRITE operation will occur if both WR and CS are pulled "low." For a more detailed timing diagram of READ and WRITE cycles, refer to the next section. pin 27 of the EPROM is used for programming of the EPROM and is usually pulled "high" for normal operation of the EPROM.

There are four RAMS cascaded together to make up a 32K-by-8 RAM for storing a complete digitized picture and the scratch pad for the CPU operation. All the RAMs and EPROM communicate with the Z-80A through the system buses. Eight bits of the data bus are used to transfer the data between the memory and the CPU. The lowest-significant thirteen bits of the address bus are used to address one of the 8192 locations inside each of the memory chips while the three-most significant address bits are used to select one of these chips. The 74LS42 decodes the addresses and enables one of the RAMs or the EPROM for a READ or WRITE operation. The 74LS42 selects one of the memories only when there is a valid address present on the address bus ($MREQ=0$).

The system memory map is shown in Figure 10. The lowest 8K of the memory is occupied by the EPROM. This is because upon power up or a RESET, the Z-80A will fetch the first instruction from location 0000H. The RAMs take up the upper 32K of the system memory. The reason for this will be explained in the next section. The rest of the memory system from 2000H through 7FFFH is not used.

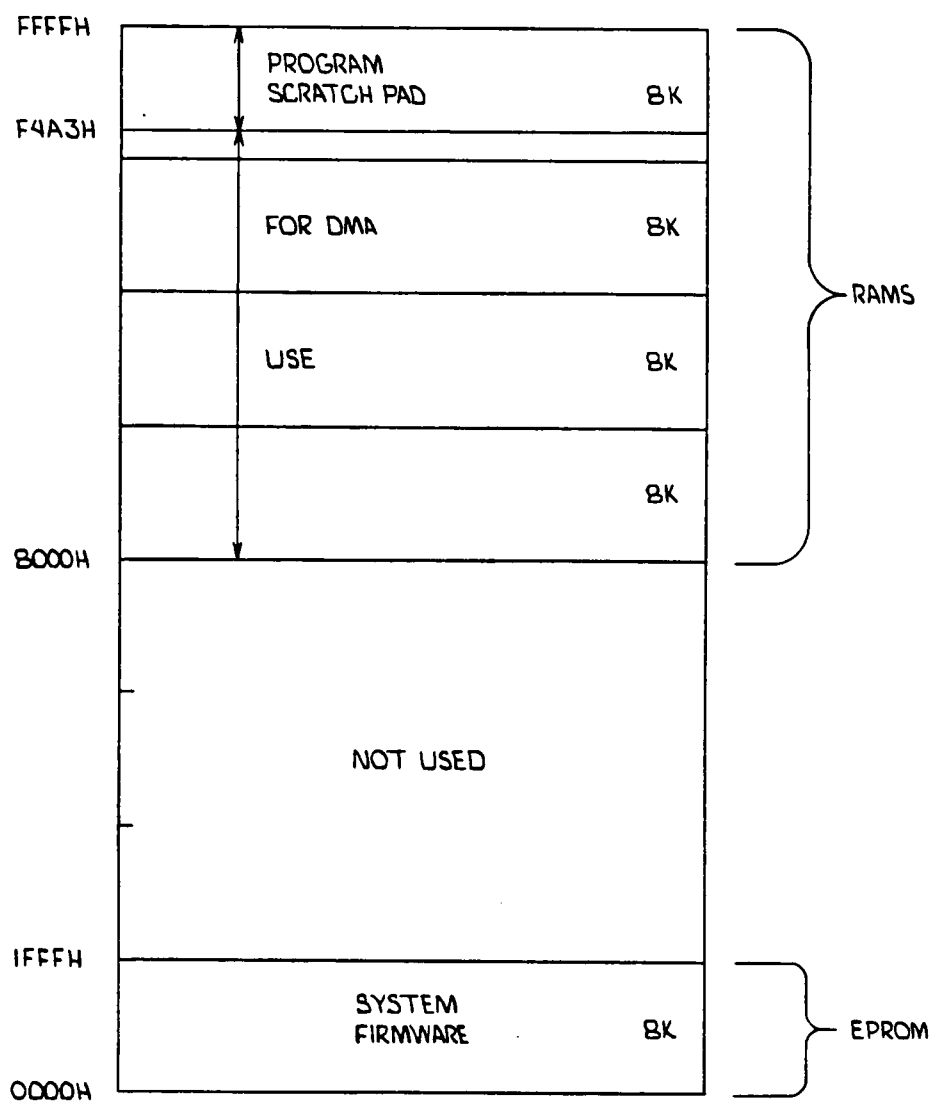


Figure 10. The System Memory Map

D. The Video Camera And Video Signal

In any image analysis system, the performance of the optical part of the system (i.e. the camera, light source, etc) affects the performance of the overall system greatly. High resolution, and easy interfacing are some of the factors which describe a good camera. The Fairchild CCD3000 video communication camera is a rugged self-contained camera which is used as the image detector of the trash-meter. The camera's high resolution (378H by 483V), inherent geometric accuracy, wide dynamic range, and reliability of a buried-channel charge-couple-device (CCD) image sensor makes it easily applied to industrial uses. The CCD-3000 provides the user with an NTSC composite video as well as all other necessary timing signals for camera interfacing. The block diagram of the camera operation is shown in Figure 11. The following signals are available from the CCD3000:

1. Composite RS-170A video signal.
2. Composite sync. signal (TTL).
3. Differential pixel clock.
4. Field index signal(TTL).

The video signal from the camera is in the standard RS-170A format. Figure 12 shows how an interlaced video signal displays on a monitor. The electron beam scans lines 1, 3, 5, 7, . . . , and 483. The electron beam then goes back to the top of the picture and starts scanning lines 2, 4, 6, 8, . . . , 482. The set of all the even or odd lines are called the even or odd fields. It takes 1/60 of a second for the electron beam to complete an

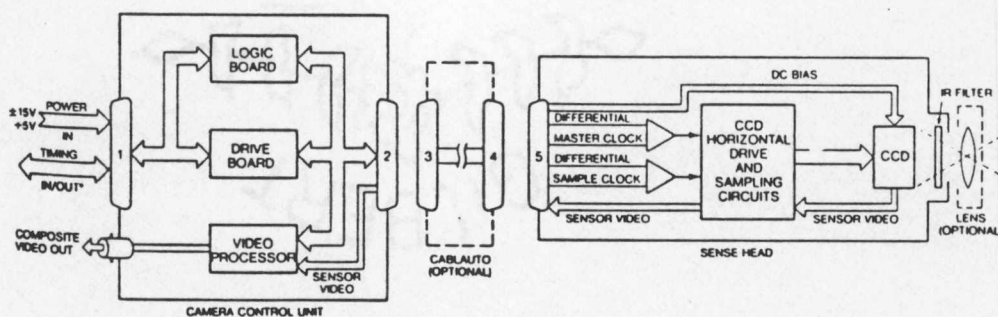
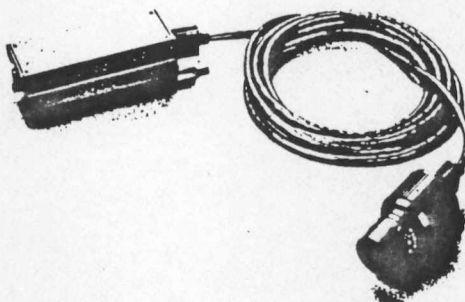
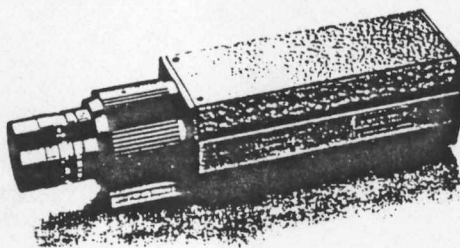


Figure 11. The Fairchild CCD-3000 Camera

— — — — — EVEN FIELD

————— ODD FIELD

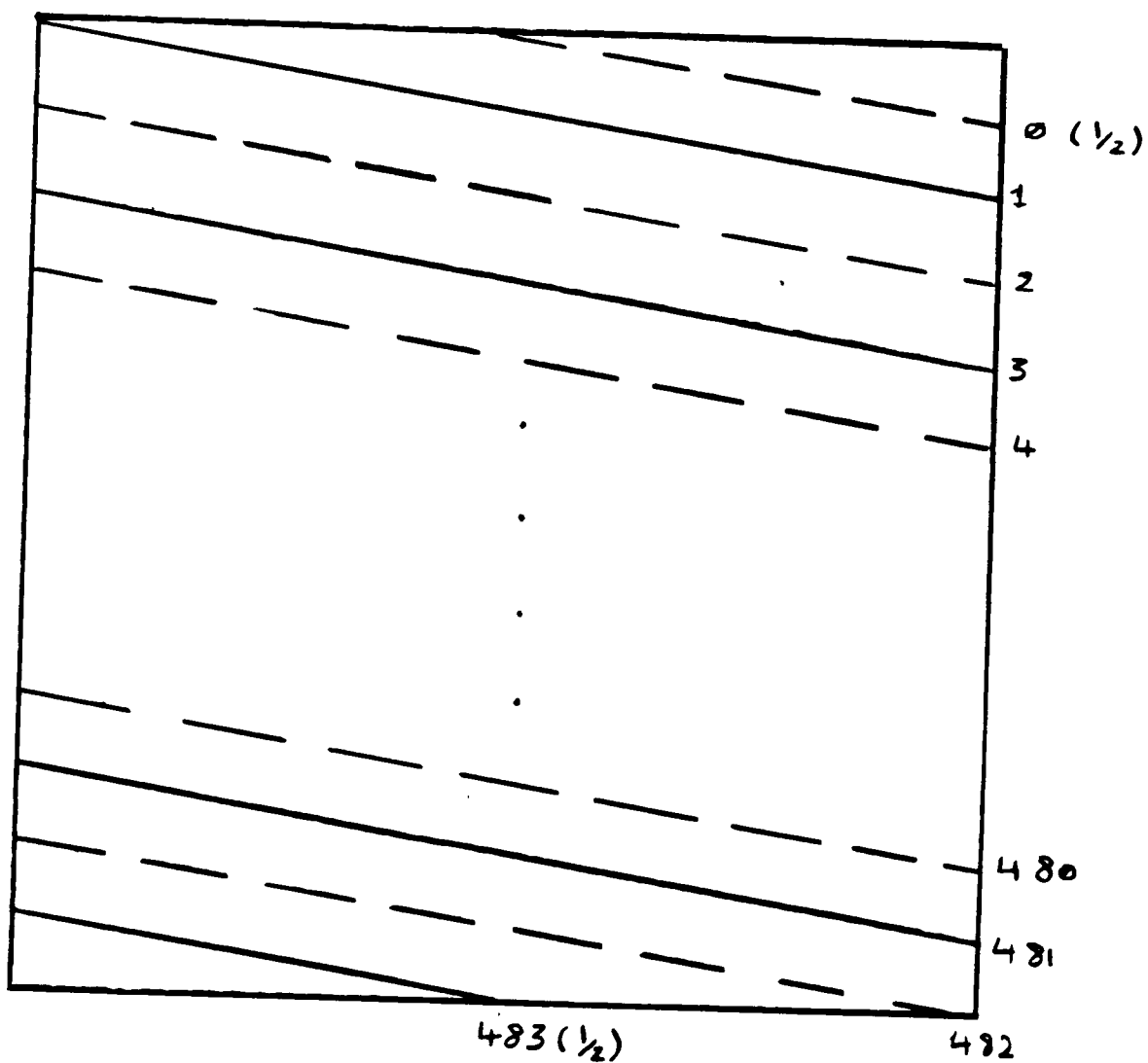


Figure 12. The Interlaced Video Format

even or odd field. One picture frame is defined as a combination of an odd field followed by an even field. Therefore, it takes $1/30$ of a second to complete one picture frame. The RS-170A scanning format is shown in Figure 13. There are 525 horizontal lines per frame. The first 21 horizontal lines of each field contains no picture information and are used merely for vertical retrace and blanking. This will leave 483 horizontal lines on the picture which contains the picture. There are 241.5 lines in each odd or even field as shown in Figure 13 (i.e. the last line in the odd field and first line in the even field are only $1/2$ lines).

The horizontal resolution of each line is a function of the camera being used. The CCD3000 has a resolution of 455 pixels (picture element) per horizontal line. The camera timing diagram is shown in Figure 14. Note that each pixel is represented by two master clock (MC) pulses. The first 77 pixels of each line do not contain any picture information. They are only used for synchronization and blanking purposes. Therefore, there are 378 valid picture pixels per scan line for the CCD3000 camera. The field index (FI) is a 600-ns TTL pulse which is output by the camera at the beginning of each odd field. The composite sync pulses are also furnished by the CCD3000. These pulses are a combination of the horizontal and vertical sync pulses which are synchronized with the composite video output by the camera.

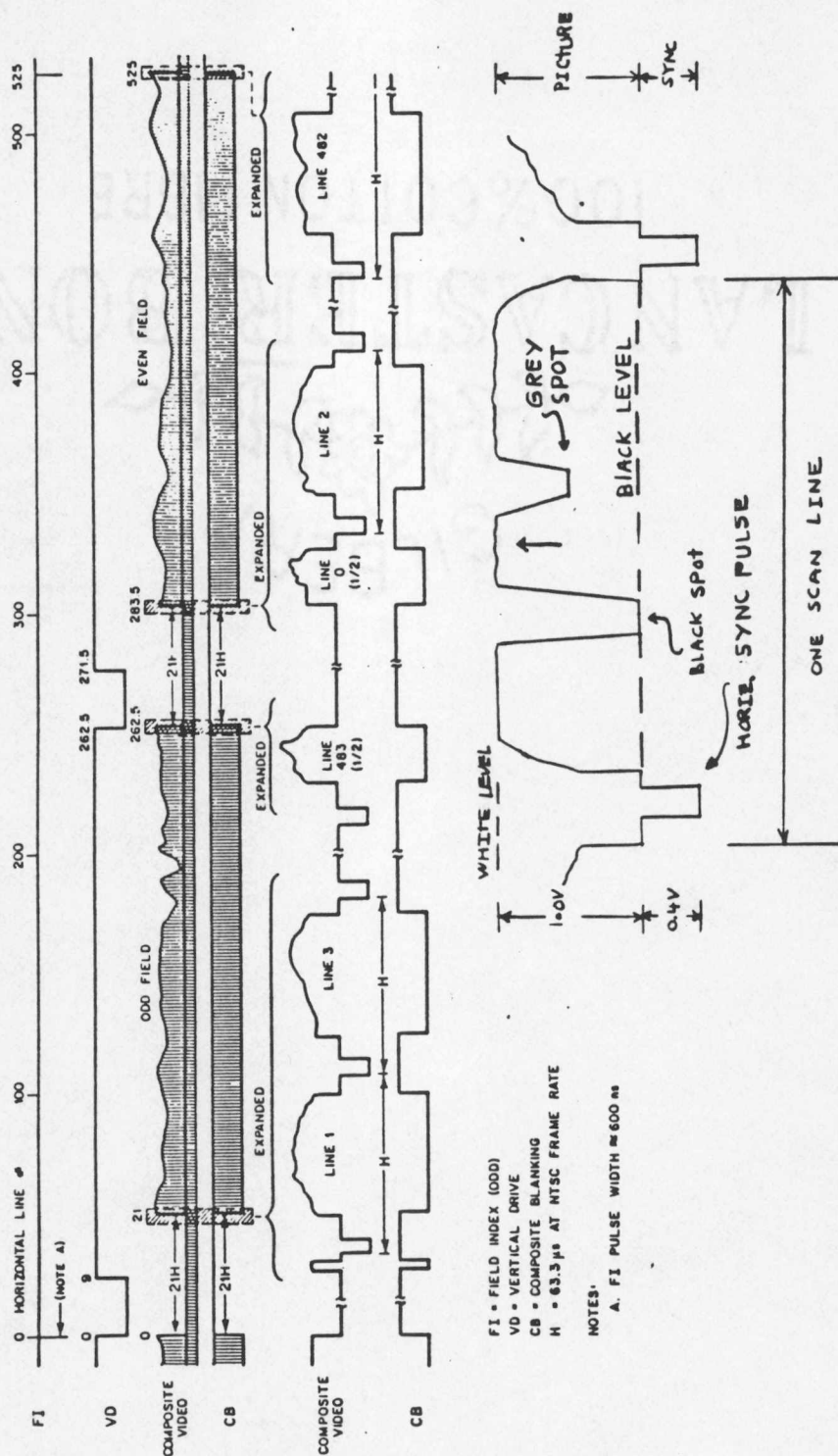


Figure 13. The RS-170 Video Format

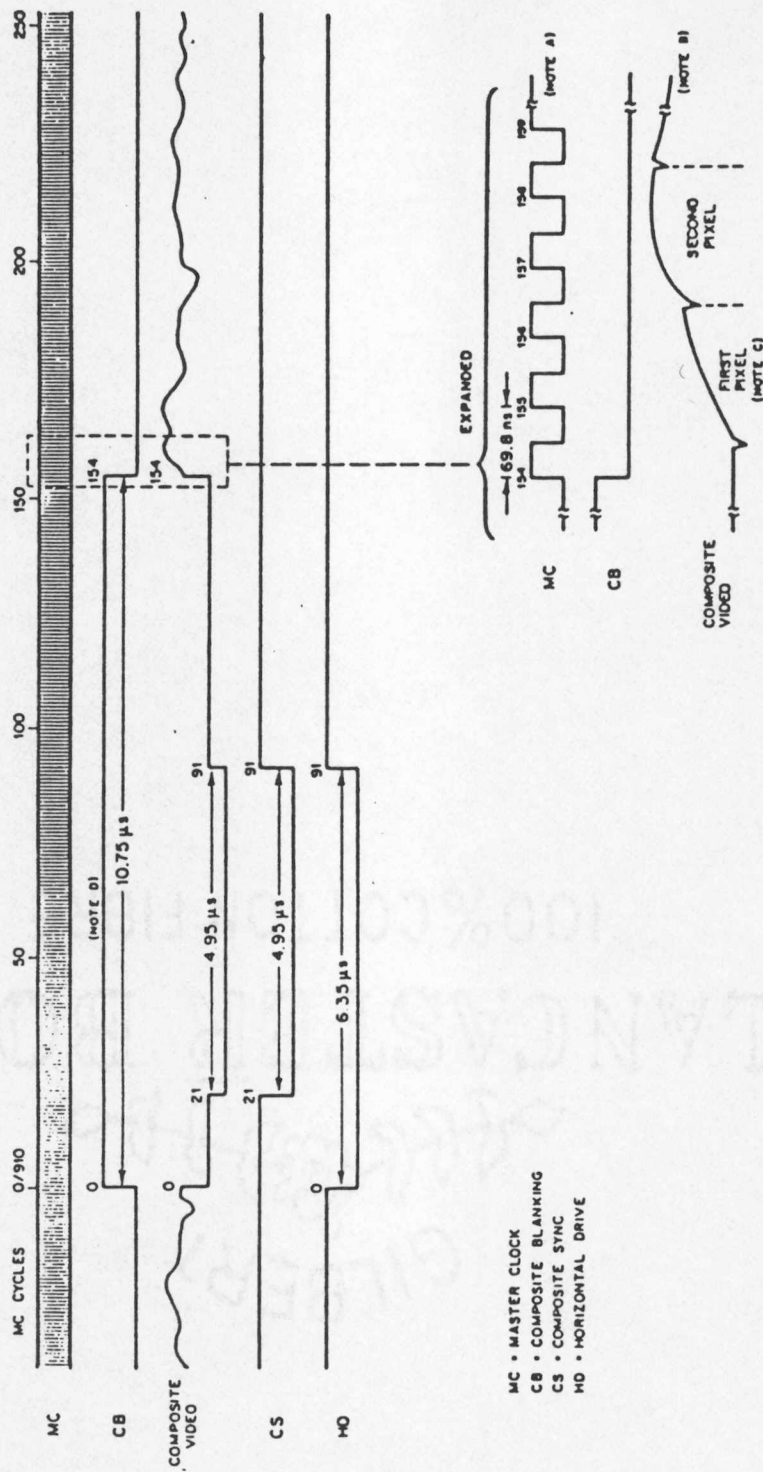


Figure 14. The CCD-3000 Timing Diagram

E. The Direct Memory Access (DMA) Controller

The DMA controller was designed to transfer serially the digitized video data to the system RAMs at the rate of the 7.15 MHz. The general block diagram of the controller is shown in Figure 15. Upon the activation of the "start" by the user, the bus request line (BUSRQ) will be brought low by the controller. Once the Z-80A recognizes the bus request and acknowledges the request by bringing bus acknowledge line (BUSAK) low, the DMA controller will take over the bus. During the DMA cycle a complete picture frame from the camera will be digitized serially, packed in an 8-bit word, and written on the system RAMs. The DMA controller derives the address, data, and control bus during a DMA cycle. Once a complete picture frame is digitized and written in to the system memory, the DMA controller will release the bus by pulling the BUSRQ line "high." As soon as the Z-80A takes over the bus, The DMA controller generates a maskable interrupt to inform the CPU that the analysis of the captured image may begin.

The actual DMA controller circuitry is shown in Figure 16 through 19. U35 is a high-speed differential voltage comparator which is used as a line receiver. R17 through R20 are used as voltage dividers.

The output of U35 is a TTL master data clock which correlates to the pixel rate as shown in Figure 19. Since there are two master data clock pulses for each pixel, the master data clock is divided down by two by means of a D flip-flop. The

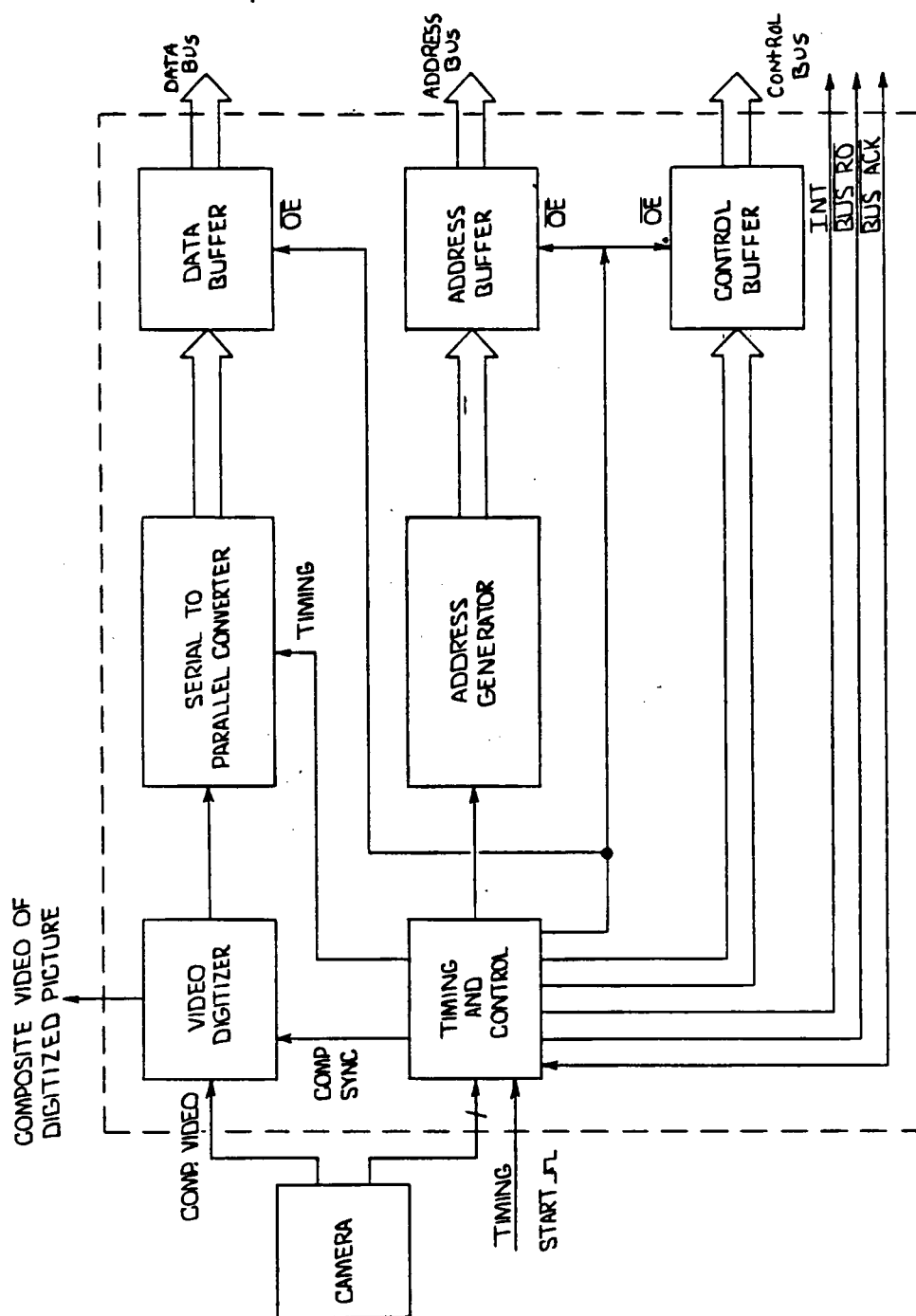


Figure 15. The Block Diagram of The DMA Controller

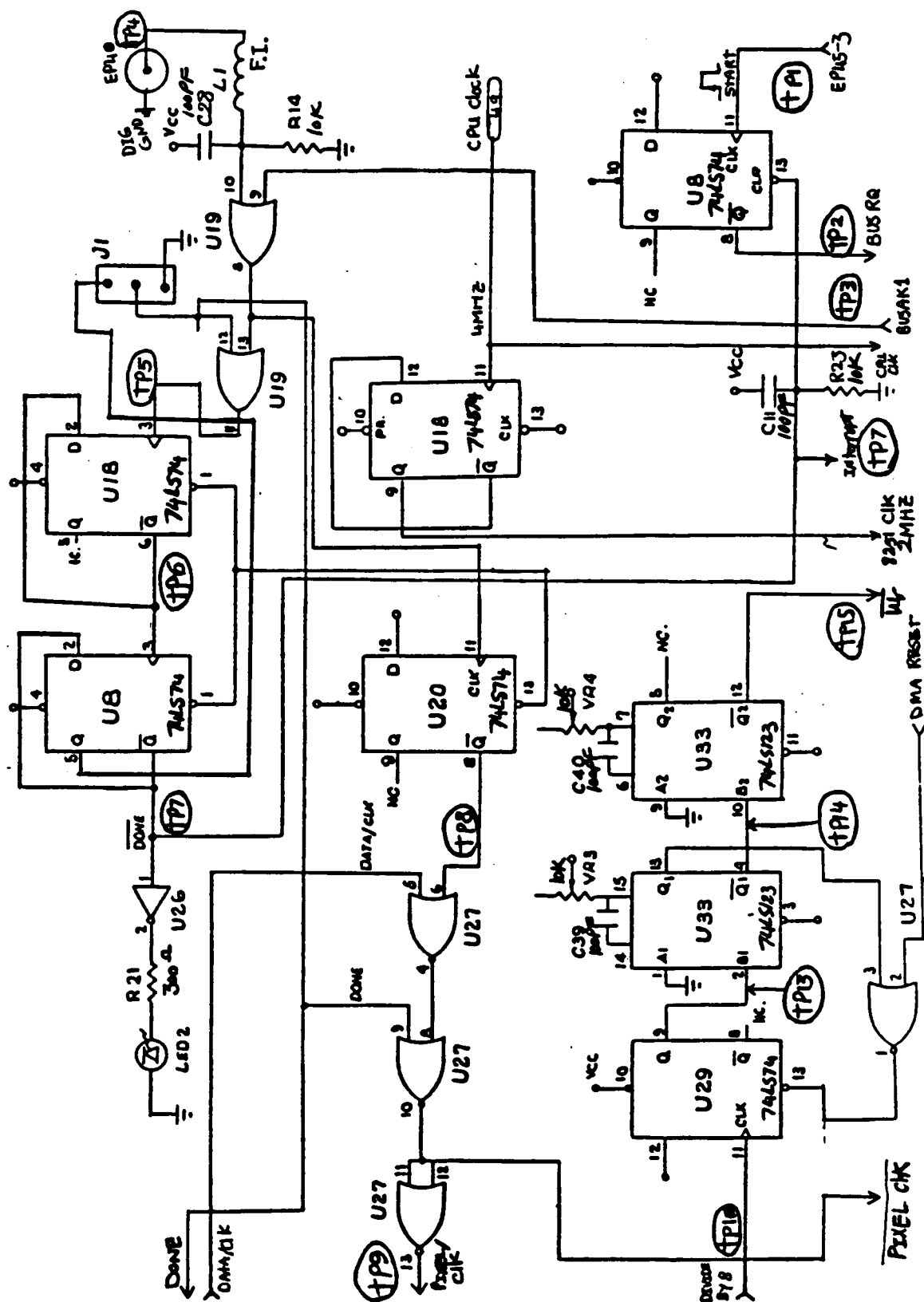


Figure 16. The Circuit Diagram of The DMA Controller Part1

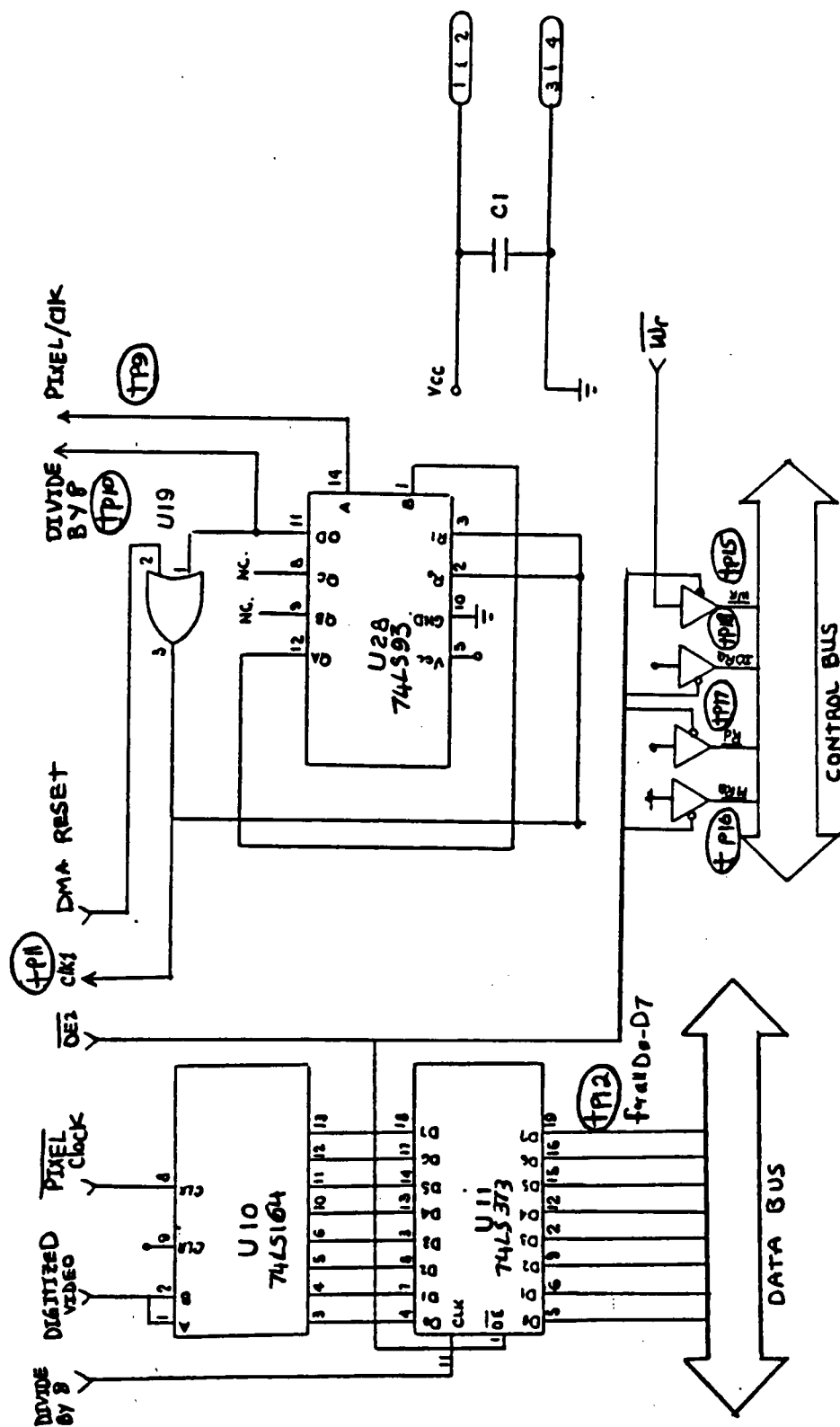


Figure 17. The Circuit Diagram of The DMA Controller Part2

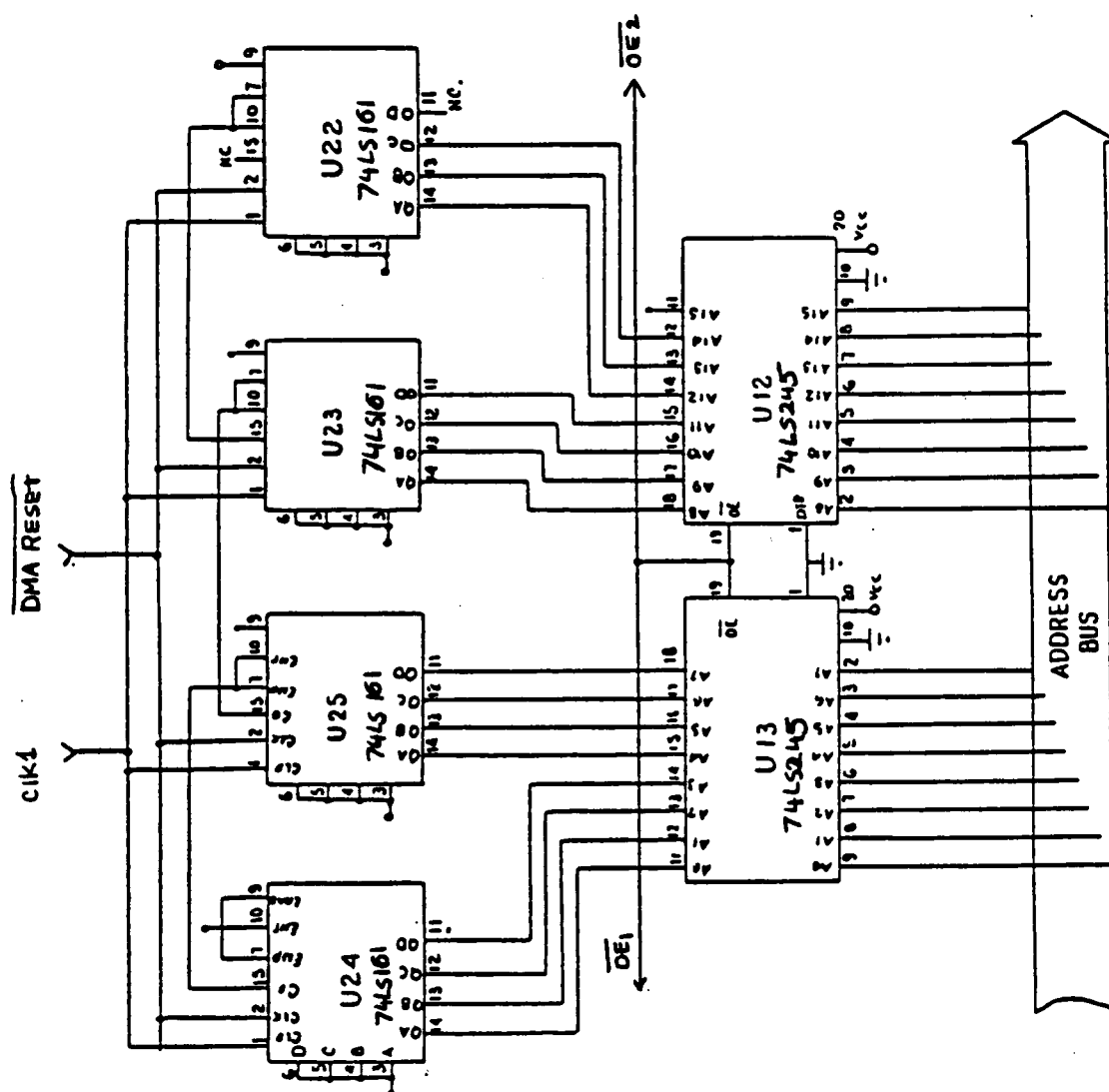


Figure 18. The Circuit Diagram of The DMA Controller Part3

composite analog video is terminated by a 75-ohm resistor for best picture quality. U34 is also a differential voltage comparator which is used as a level detector. A 6.95V high precision zener diode is used to establish a stable threshold trash reference (i.e. how dark the trash has to be in order to be considered trash). The voltage level of the incoming video signal is compared with the threshold level. This threshold voltage is a function of the optical system (i.e. the brightness of the illuminating light, the iris setting of the camera, etc). The threshold level is usually set at a level at which a clear differentiation between cotton (white level) and the trash on the cotton (black level) is produced. Pin 6 and 7 of U34 are complement TTL output of the comparator. The - output is used for the construction of the digitized video. This signal is mixed with the composite sync signal from the camera to form a composite video signal of the digitized picture which could be displayed on any NTSC monitor.

As explained before, Field Index (FI) is a 600-nsec pulse which occurs at the beginning of each odd field. Upon a "start" signal from the user, U8 gets set and this will generate a bus request to Z-80A. The Z-80A acknowledges this request by bringing the BUSAK "low." This will enable the field-index to propagate through the first NOR gate. Assuming that a master reset had been applied to the system right after the power up, the output of the U8-8 would be "low." This will also enable the field index to penetrate through the second NOR gate. Two D-type flip-flops U18 and U8 are cascaded together to form a 2-bit

counter. This 2-bit counter counts up the number of field index pulses which pass through the NOR gates. The timing diagrams for all above sequence are shown in Figure 20.

After activation of the "start," the first field index pulse initiates two parallel processes. The first process is that the 2-bit counter starts counting the FI pulses and stops counting as soon as the scanning of the first frame is done. This counter counts up two field index pulses and stops counting and also generates an interrupt to the Z-80A. This interrupt informs the Z-80A and the rest of the system hardware that a complete picture frame has been digitized and stored in the RAMs. The same interrupt signal is connected to the clear input of the U8-13 which will clear the flip-flop causing the BUSRQ input to be pulled high so the Z-80A can start analyzing the picture. The very first field index pulse will also set the U20 D-type flip-flop. This will enable the data clock to propagate through the NOR gate to the 3-bit counter (74LS93) and clock input of a shift register(U10).

The positive digitized video is fed to the shift register (U10) to be packed as an 8-bit word. The shift register shifts the incoming data at the data clock frequency (i.e. pixel rate) which is equal to

$$\text{DATA CLOCK FREQUENCY} = \text{MASTER DATA CLOCK FREQUENCY} / 2 = 14.3\text{MHz} / 2 = 7.15\text{MHz}$$

The counter (74LS93) is used as a divide by eight counter and counts up any time a pixel is shifted into the shift register. After counting 8 pixels, the counter generates a pulse to show that the shift register is full with 8 bits of data. The 8-bit data in the shift

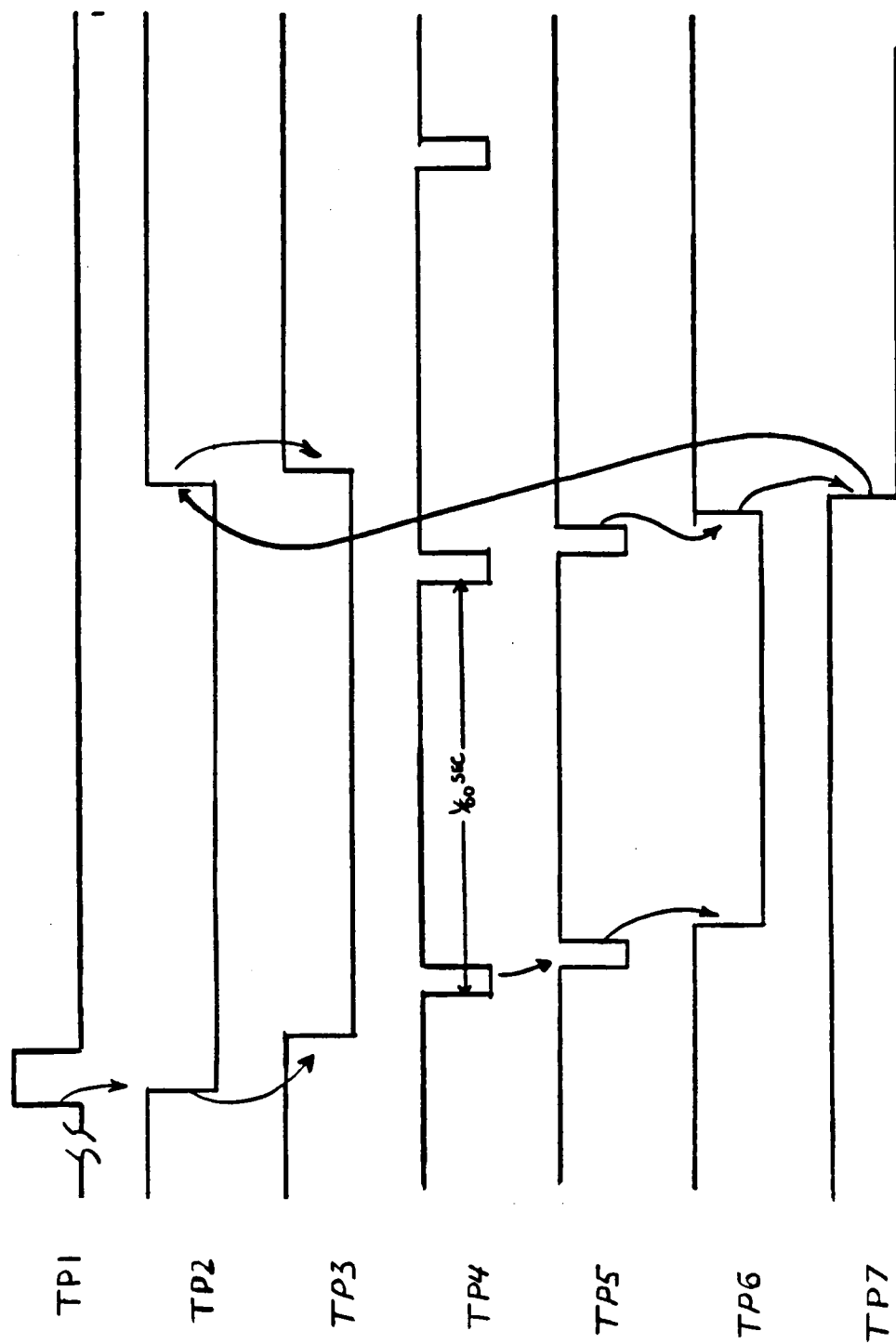


Figure 20. The Timing Diagram 1

register are now ready to be written in to the RAMs. The timing diagrams for the above sequence are shown in Figure 21. The maximum writing access time for the RAMs is 150 nsec. This means that the data in the shift register have to be stationary for at least 150 nsec. However, this is not possible because the next pixel is going to be shifted in to the shift register in only 140 nsec (i.e. shift register is shifting at rate 7.15MHz continuously). Therefore, an 8-bit latch (U11) is used to store the data from the shift register for writing set-up time. U11 is a 74LS373 with tri-state output which could interface with the system data-bus without any extra hardware. After counting the 8th pixel going in to the shift register, U28 will generate an approximately 50-nsec pulse which clocks the data from the shift register into the latch.

Four 4-bit counters (74LS161) are cascaded together to form a 16-bit binary counter. This 16-bit counter always points to a memory location in the RAM where data from the latch has to be written. After writing the first 8 bits of the data, the counter is incremented by the same signal which clocks the second set of 8-bit data into the latch. There are 525 horizontal lines per TV frame and each line contains 455 pixels. Therefore total number of pixels written in to the RAMs is

total number of pixels = $525 \times 455 = 238,875$

and there is one memory location used for eight pixels so

total number of memory locations used = $238875 / 8 = 29859 = 29K$

This means that the 16-bit counter counts from 0000H to 74A3H. The counter is interfaced to the system address bus through two

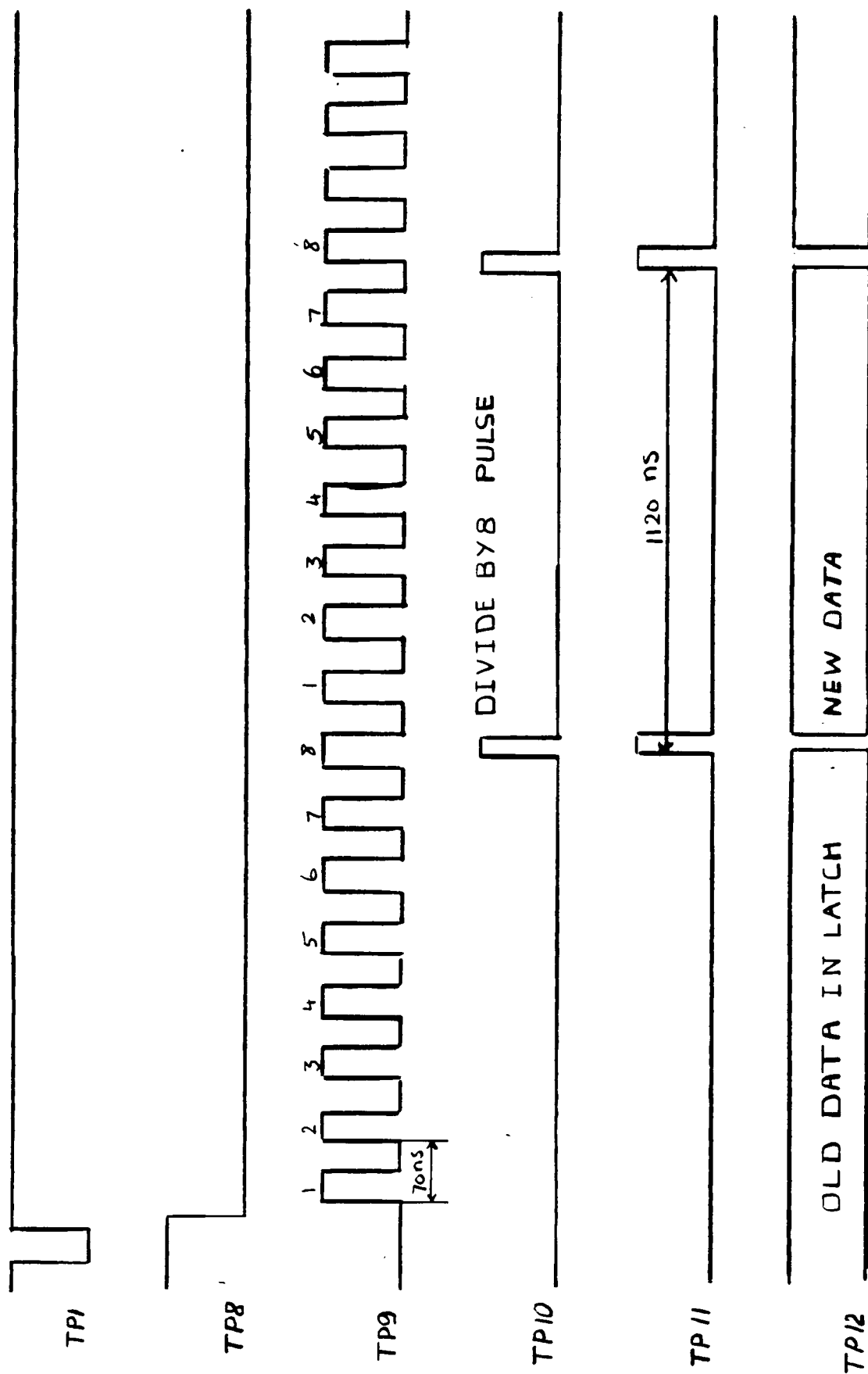


Figure 21. The Timing Diagram 2

cascaded 8-bit bus driver with tri-state output (74LS245). Since the lower 8K of the memory map is occupied by the EPROM , the upper 32K of the memory (i.e. 8000H through 0FFFFH) is allocated to the RAMs and for DMA access. The DMA controller accesses the upper 32K of the memory simply by having the most significant of the address (MSB of the bus driver input A15) set to logic "high." This causes the DMA controller to write a complete picture frame into locations 8000H through F4A3H.

After the DMA controller furnishes the address and the data for the RAMs, it waits for a period of time for address and data set-up purposes. After this delay then it will generate a write pulse to the RAMs to complete the write cycle. There are two one-shots (U33) cascaded together to generate the time delay and the write pulse. Figure 22 shows the timing diagram involved to produce a "write" pulse. 74LS123 (U33) is a dual one-shot whose output pulse width is controlled through an external capacitor and resistor. The first one-shot is used to generate the time delay for the data and address set-up time. The other one-shot fires the actual write pulse as soon as the time delay is over. The same pulse which increments the address counter and clocks the data latch is used to trigger the time delay one-shot. The length of this pulse may be even less than 40 nsec which exceeds the lower limit of the one-shot input pulse width. Therefore, a D-type flip-flop (U29) is used to make sure that every pulse is seen by the time delay one-shot. The output of the D flip-flop (U29-9) triggers the time delay one-shot as soon as the flip-flop

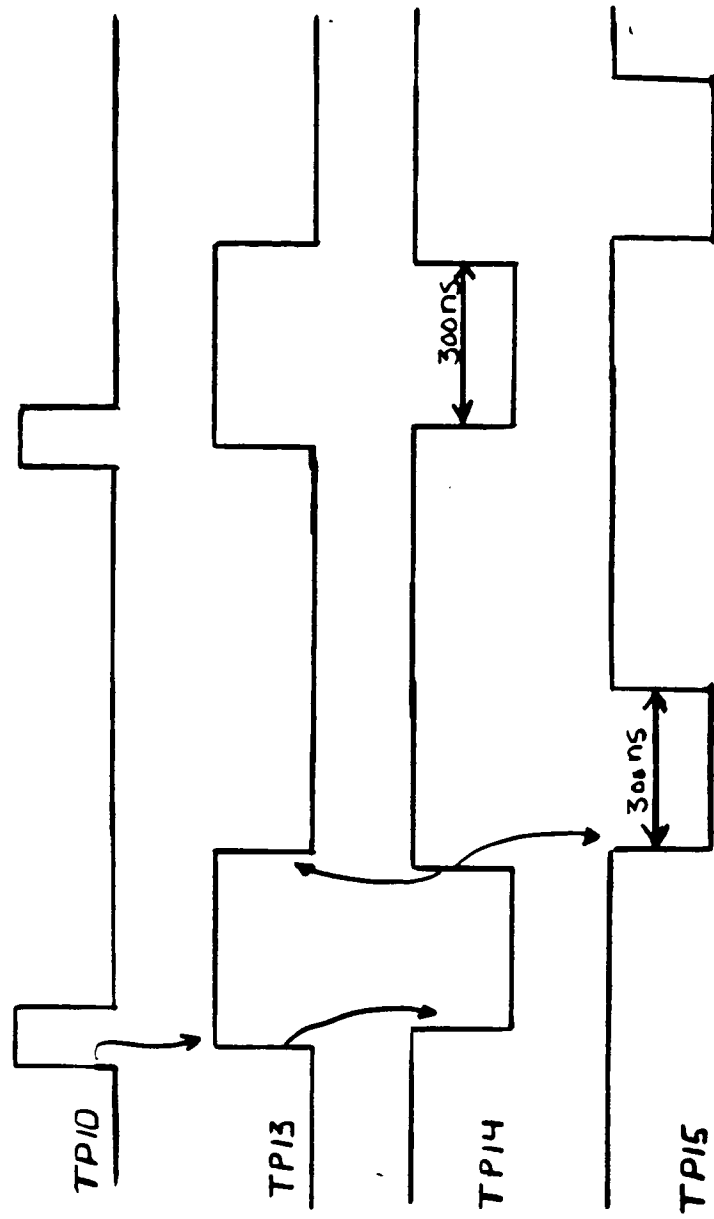


Figure 22. The Timing Diagram 3

is set by the pulse from U28-11. The time delay one-shot then generates two complement pulses. The Qnot output is used to trigger the write pulse one shot while the Q output resets the D flip-flop mentioned earlier. The output of the write pulse one-shot is interfaced to the write signal on the system bus through a tri-state buffer (U15).

During the DMA cycle, the DMA controller is supposed to drive the read (Rd), I/O request (IOREQ), and memory request (MREQ) as well as data and address buses (Figure 23). Since there are not any I/O or read operations during the DMA cycle, both IORQ and Rd lines on the system bus are driven "high" through two tri-state buffers. For simplicity, the system memory request is driven "low" so that the address decoder can enable one of the memory chips as soon as the address counter is incremented. The system memory-request line is also driven by the DMA controller through a tri-state buffer. The output-enable controls of the address (U12, U13), data (U11), and control (U15) bus drivers are all connected together. The bus-acknowledge from the Z-80A is run through two inverters for buffering and polarity purposes and connected to the output control of the DMA bus drivers.

There are two different resets used in the trash-meter. One of them is the user push-button (master reset) reset which is initiated by the user. This reset is an active-low signal and runs in to the trash-meter card through the STD bus system reset. Bringing this signal "low" momentarily will cause a reset to the CPU, USART, counters and the flip-flops. The other reset (DMA reset) is initiated

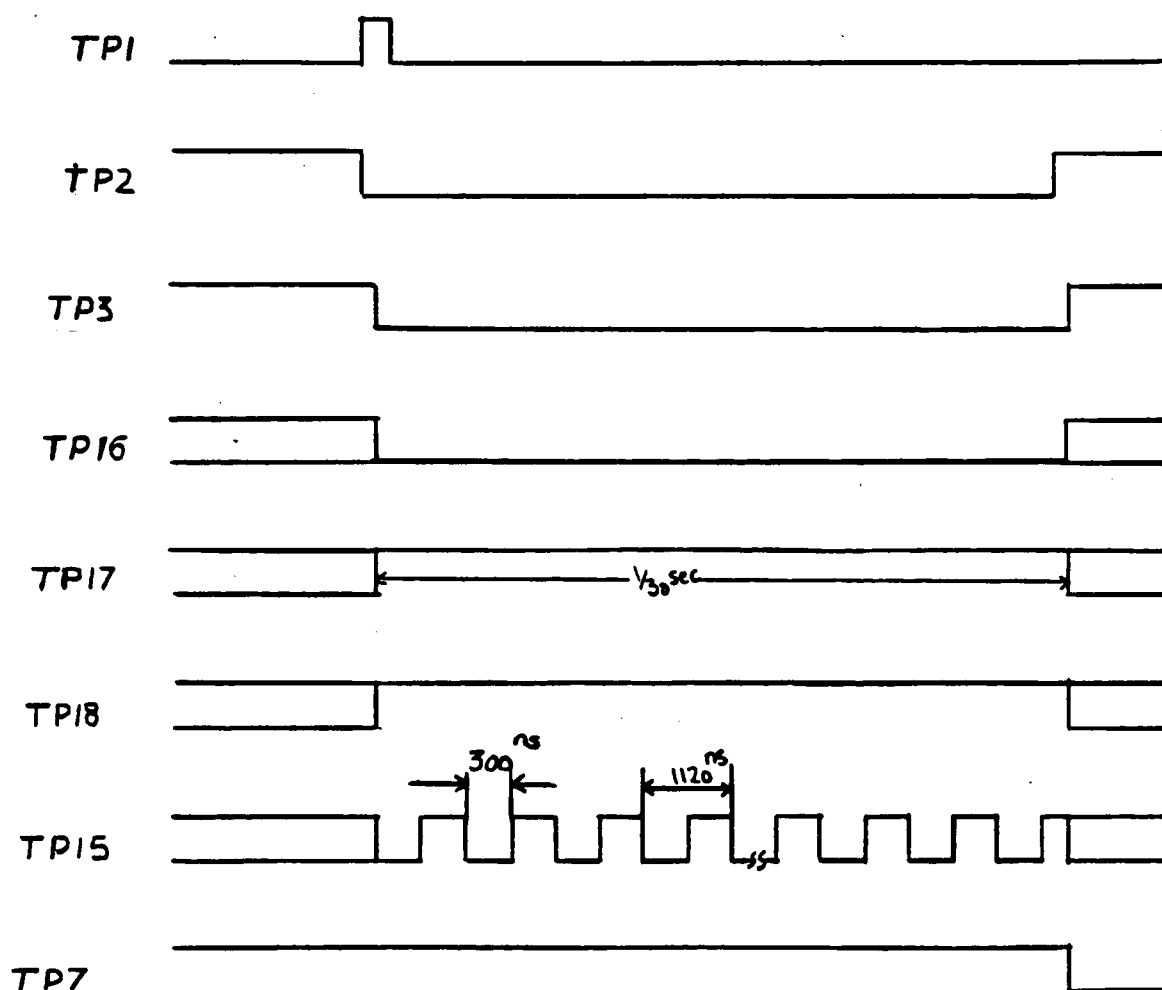


Figure 23. The Timing Diagram 4

by the software and only resets the DMA circuitry (counters and the flip flops). The DMA reset is initiated any time the 8251A is selected by software. The push-button reset is usually applied after power-up to make sure that the system hardware has been reset properly. Once a complete picture frame has been analyzed and transferred to the host, all the counters and flip-flops need to be reset before another picture frame may be analyzed.

The format by which a picture is partitioned and written into the RAMs is shown in Figure 24. The first part of the RAM is occupied by vertical pulses for the odd field which has been digitized and stored along with the rest of actual picture pixels. The vertical pulses contain no picture informations and has a length of 21 scan lines. The next part of the RAM is taken up by 241.5 lines of the odd field. The last two parts are used by the vertical pulses (21 lines long) and 241.5 lines of even field. Appendix B shows the actual content of the RAMs when a white sheet of paper is digitized and stored. Memory locations 8000H thru 84AAH belongs to vertical pulses for the odd field. Memory locations 84ABH thru BA51H belong to the scan lines of the odd field. The first scan line starts at location 84ABH. Note that the first 77 pixels are actually the horizontal sync pulses which have been digitized right before the picture pixels. The actual 378 picture pixels are in locations 84B4H (only the three-least significant bits) thru 84E3H (only the seven-most significant bits). Line one is followed by the horizontal sync pulses of the third scan line. The

	Scan Line 482
	Scan Line 2
	Scan Line 0 (1/2 Line)
BA52H	Vertical Pulses For The Even Field
	Scan Line 483 (1/2 Line)
	Scan Line 481
84E3H	Scan Line 3
84ABH	Scan Line 1
8000H	Vertical Pulses For The Odd Filed

Figure 24. The Format of The Digitized Video In The Rams

vertical pulses and scan lines for the even field have the same format as the odd field.

F. Miscellaneous

As mentioned before, the trash-meter is built into a Spinlab-900 high-volume test instrument. The 900 testing instrument requires the trash-meter to be capable of multiplexing two video signals. One of the video signals is the digitized picture of what the trash-meter is seeing. The other signal is an external video signal which comes from outside as shown in Figure 19, p35. In the Figure 19, p.35, the external video is referred to as "monitor in" and "monitor cs" is the composite-sync pulses associated with this noncomposite video signal. U16 is a quad two-to-one multiplexer with non-inverting outputs. One multiplexer is used to switch the digitized video from the camera from the external video signal (both non-composite TTL video). Another multiplexer is used to select either the composite sync pulses from the camera or the external video circuitry. The output of these two multiplexers is a non-composite video and a composite sync pulses. These outputs are then mixed together to form a composite video which may be displayed on a standard monitor. This signal is shown in Figure 19, p35, as "monitor out."

CHAPTER II

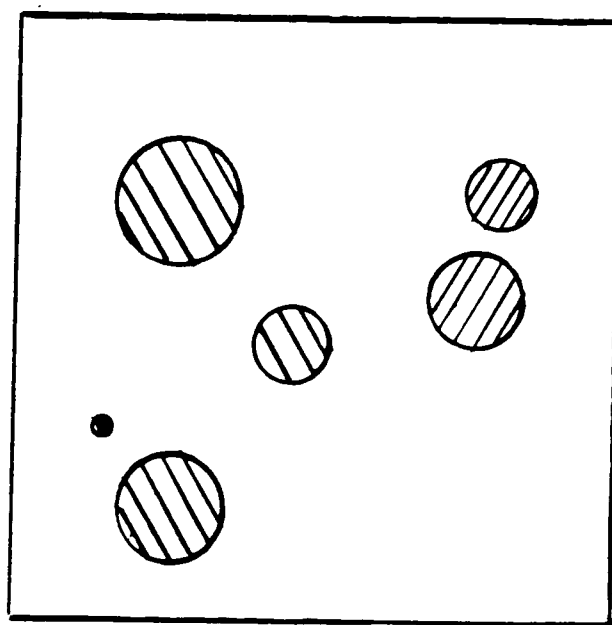
SOFTWARE DESIGN

A. Algorithm

There are two different parameters which the trash-meter computes. The first parameter is called the trash area defined as the area of the cotton surface which is taken up by the trash. The other parameter is trash count which represents the total number of distinct trash spots present on the surface of the cotton. This is a useful parameter which can help to differentiate between two different grades of cotton having the same trash area and different trash count as shown in Figure 25.

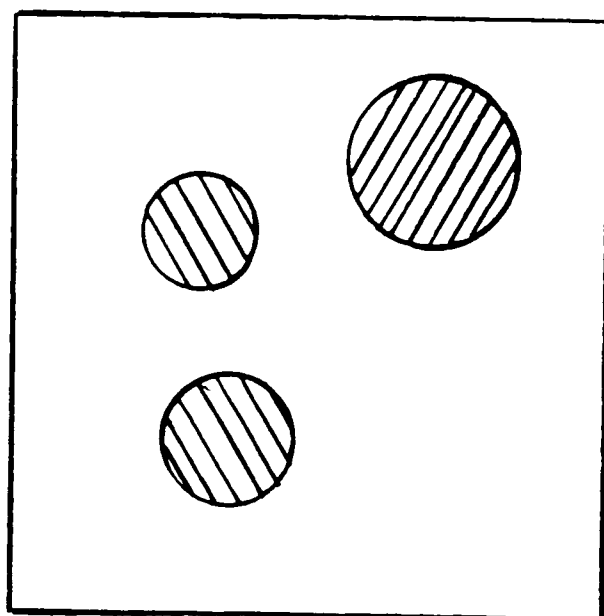
A picture of cotton is partitioned into a 482 by 378 matrix as shown in Figure 26 (ignoring two 1/2 lines). Each pixel represents either a white (cotton) or black (trash) spot. The area of the trash may be calculated by counting the total number of trash pixels (black).

Another desired parameter is the total number of distinct trash spots. The problem then narrows down to how to group the trash pixels together in order to form a trash spot. Figure 27 shows a more detailed picture of Figure 26. Take two consecutive TV scan lines and call them "reference" and "data" lines for clarity purposes. As long as a trash spot on the "data" line is adjacent to another trash spot on the "reference" line, these trash spots belong to same group (i.e. they are both a part of same trash spot). For example, there are two trash spots on the "reference" line as shown in Figure



TRASH AREA 13.9%

TRASH COUNT 6



TRASH AREA 13.9%

TRASH COUNT 3

Figure 25. The Trash Area And Trash Count of Cotton

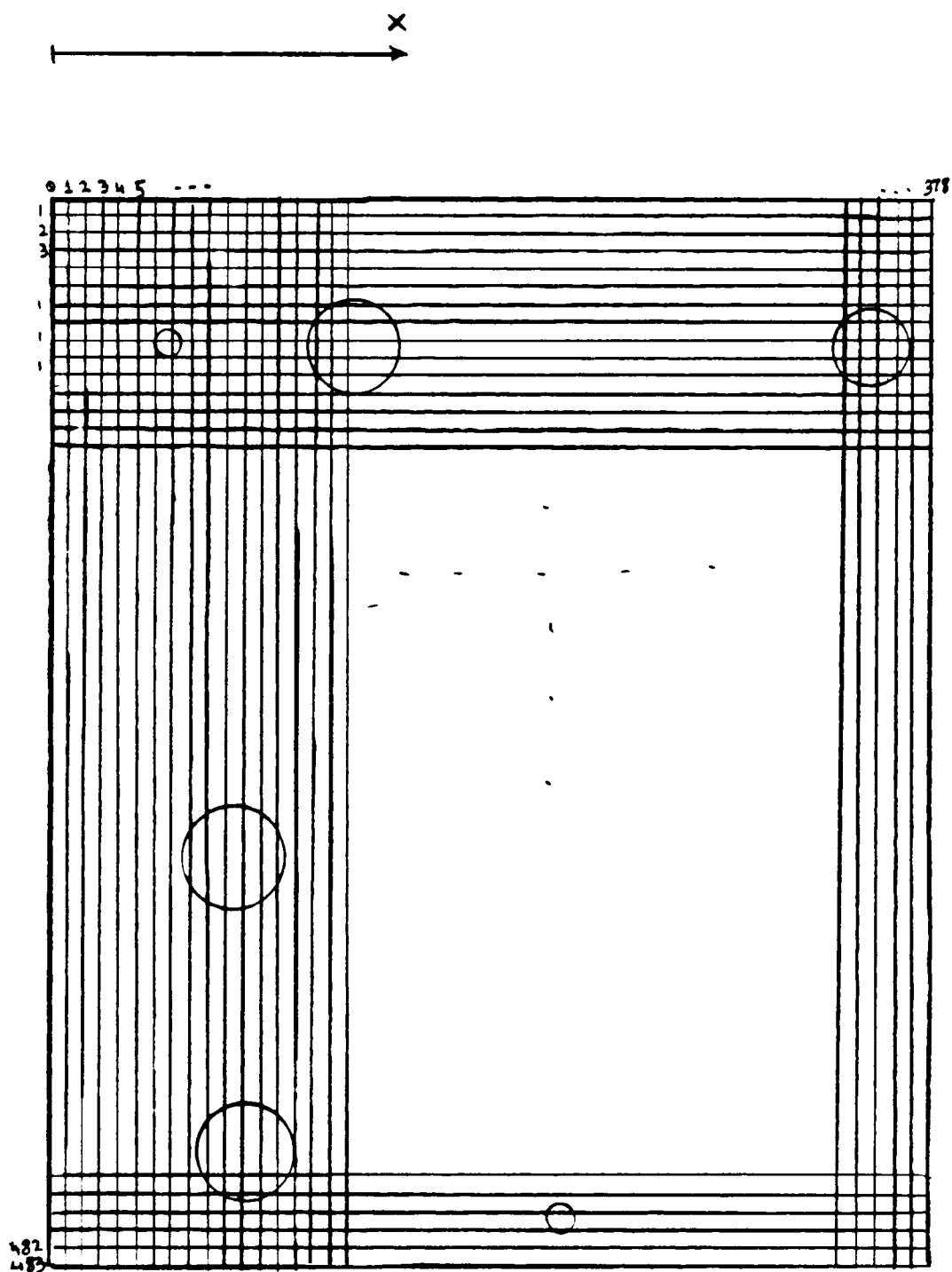


Figure 26. The Partitioning Format of The Cotton Image

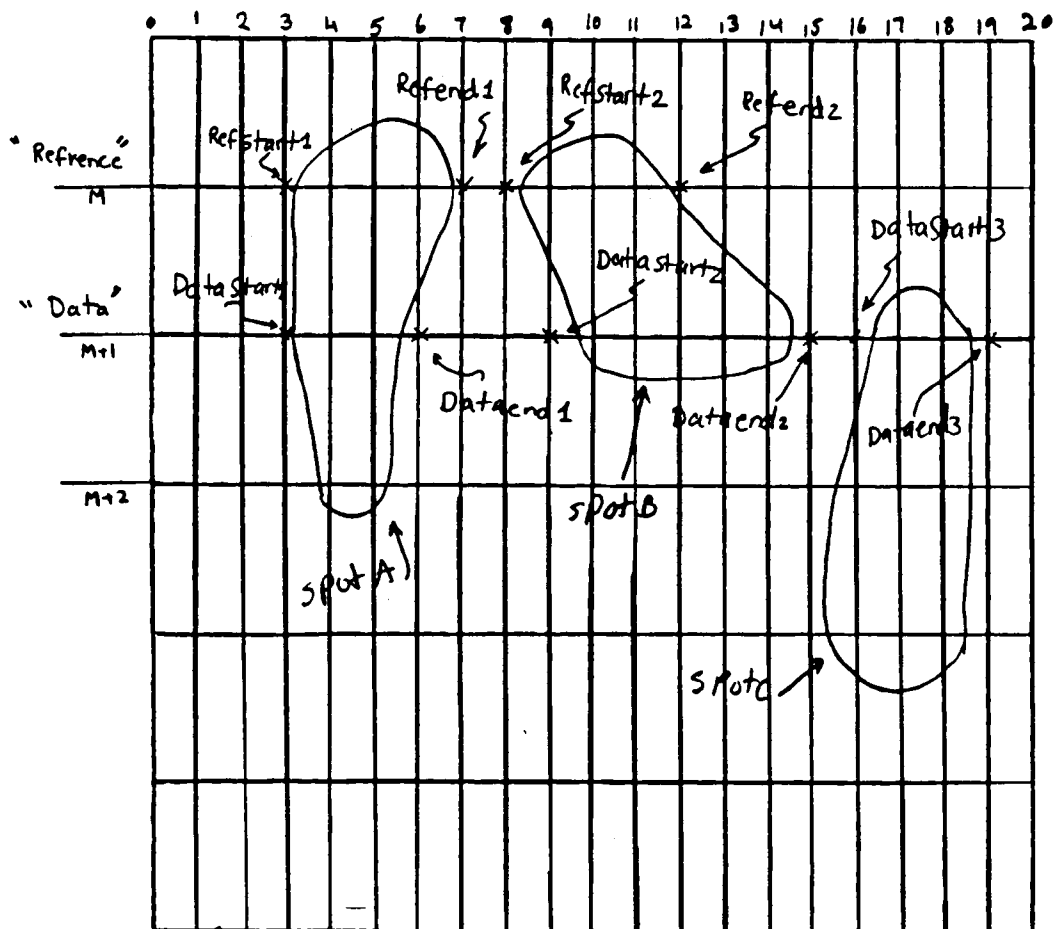


Figure 27. The Trash Spots And Scan Lines Part1

27. These trash spots have starting and ending X-coordinates which are referred as "refstart" and "refend." The same thing is true for the "data" line and they are called "datastart" and "dataend." If any of the following inequality holds between any two "data" and "reference" trash spots then the trash spots on the "reference" and "data" lines belong to same trash group.

$$\text{REFSTART} \leq \text{DATASTART} \leq \text{REFEND} \quad (1)$$

$$\text{REFSATART} \leq \text{DATAEND} \leq \text{REFEND} \quad (2)$$

For example, for a given cotton picture with some trash spots like in Figure 27 :

refstart1= 3	datastart1= 2
refend1=7	dataend1= 6
refstart2=8	datastart2= 9
refend2= 12	dataend2= 15
Total number of trash spots on the reference line = 2	datastart3= 16
	dataend3= 19
	Total number of trash spots on the data line = 3

Starting with the first trash on the data line :

$$\text{REFSTART1}=1 > \text{DATASTART1}=2 < \text{REFSTART1}=7$$

$$\text{REFSTART1}=1 < \text{DATAEND1}=6 < \text{REFEND1}=7$$

So the inequality (1) holds true which shows that the trash spots number one on the "reference" and "data" lines belong to trash group A(spotA). The second trash spot on the "data" line yeilds the following inequalities:

$$\text{REFSTART1}=3 < \text{DATASTART2}=9 > \text{REFEND1}=7$$

$$\text{REFSTART1}=3 < \text{DATAEND2}=15 > \text{REFEND1}=7$$

$$\text{REFSTART2}=8 < \text{DATASTART2}=9 < \text{REFEND2}=12$$

Therefore the trash spot number two on the "data" line and the "reference" line belong to same group (i.e. both belong to spot B). Comparing the third trash spot on the "data" line with the ones on the "reference" line results :

$$\text{REFSTART1}=3 < \text{DATASTART3}=16 > \text{REFEND1}=7$$

$$\text{REFSTART1}=3 < \text{DATAEND3}=19 > \text{REFEND1}=7$$

$$\text{REFSTART2}=8 < \text{DATASTART3}=16 > \text{REFEND2}=12$$

$$\text{REFSTART2}=8 < \text{DATAEND3}=19 > \text{REFEND2}=12$$

Since there are not any more trash spots on the "reference" line for comparison and none of the inequalities (1) and (2) is satisfied, trash spot 3 does not group with any of the trash spots on the "reference" line. Therefore this trash spot belongs to a new group (i.e. group C). Since there are not any more trash spots on the "data" to analyze, one can define this line as a new "reference" line and consider the next adjacent scan line as a new "data" line and follow the pocedure just explained as shown in Figure 28.

The analysis takes place through the following steps:

1. Set the TV line pointer, TRASH-AREA, TRASH-COUNT to zero.
2. Increment the TV line pointer by one.
3. The analysis is completed if TV line pointer is greater than 482.
4. Define the line which TV line pointer pointing to as a "reference" line and check for trash pixels.

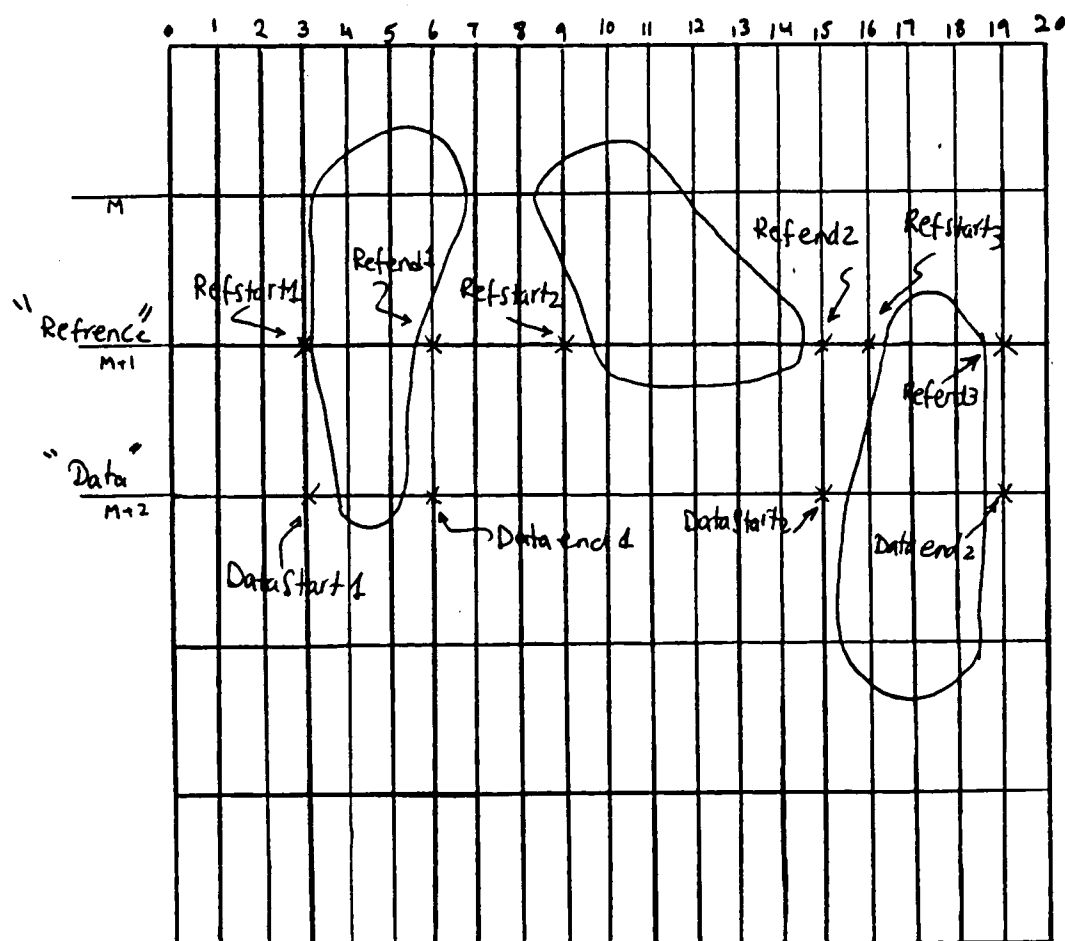


Figure 28. The Trash Spots And Scan Lines Part2

5. If there are not any trash pixels, go to step 2.
6. Compute the number of trash pixels (TRASH), the number of trash spots on the line (NREF), and the starting (REFSTART) and ending (ENDREF) coordinates of all the trash spots on the "reference" line.
7. Increment the trash area and trash count according to the number of TRASH and NREF on the "reference" line.
$$\text{trash area} = \text{trash area} + \text{TRASH}$$
$$\text{trash count} = \text{trash count} + \text{NREF}$$
8. Increment the TV line pointer by one.
9. If the TV line pointer is greater than 482, the analysis is completed.
10. Define the line which TV line pointer pointing to as a "data" line and check for trash pixels on it.
11. If there are not any trash pixels on the line, go back to step 2.
12. Compute the number of trash pixels (TRASH), the number of trash spots (NDATA), and starting (DATASTART) and ending (ENDDATA) coordinates of the all trash spots on the "data" line.
13. Increment the trash area by
$$\text{trash area} = \text{trash area} + \text{TRASH}$$
14. Compare the trash spots on the "data" line with the trash spots on the "reference" line to see if any of the trash spots on the "data" line may be grouped with any in the "reference" line as described in previous paragraph).

- 15) The trash spots on the "data" line which are not grouped with any of the trash spots on the "reference" line, will be the new trash spots. Therefore the trash count must be incremented by the number of new trash spots on the "data" line (NNEW).

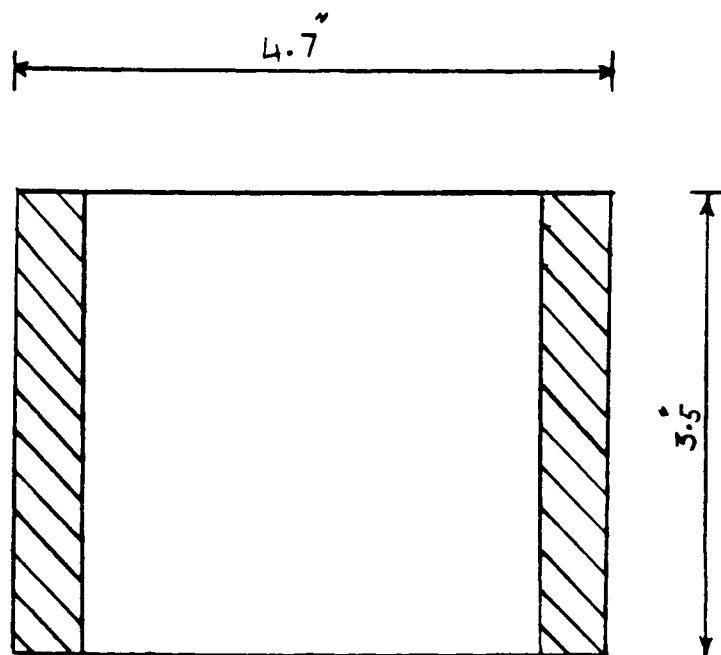
trash count = trash count + NNEW

16. Define the "data" line which just been processed as a new "reference" line.

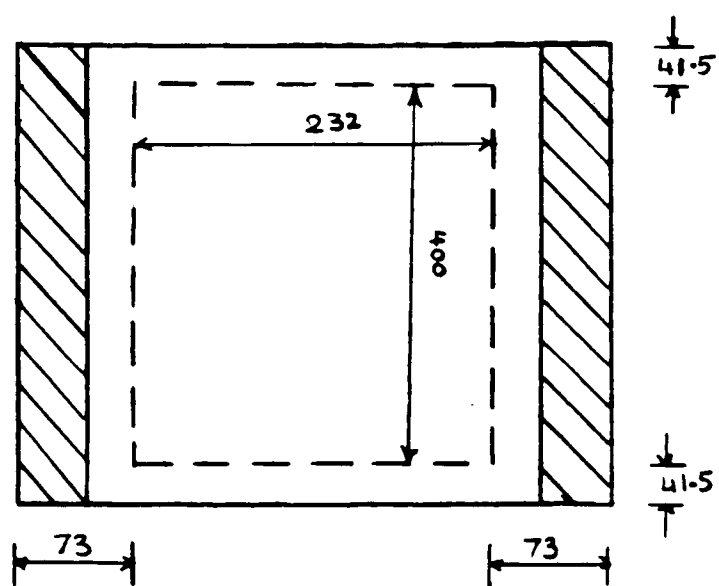
17. Go to step 8.

The observation of the trash meter is a 3.5" by 3.5" square. The aspect ratio of the video camera is 4:3 (Horizontal:Vertical). Therefore, the camera field of view has to be 4.7" by 3.5" (Horizontal:Vertical) in order to cover up only the window. This is shown in Figure 29a. Since it is difficult to set the camera so that it will cover exactly 4.7" by 3.5," a software window is applied to the picture; the main analyzing routine only analyzes what is inside of the dashed window as shown in Figure 29b.

By trial-and-error the author has found out that ignoring 41.5 horizontal lines from top and bottom and 73 pixels from left and right of each line yield a 232 by 400 (horizontal:vertical) picture matrix which solves all the above problems yet leaves a very good resolution. For example the first scan line of the odd field starts at location 84ABH as shown in Figure 30. The first 77 pixels are the horizontal sync pulses occupying the first $9 \frac{5}{8}$ bytes (i.e. each byte = 8 pixels). The picture pixels start right after the sync pixels as shown in Figure 30. Ignoring the first 73 picture pixels



(A)



(B)

Figure 29. The Camera Observation Window

	Scan Line 482
	Scan Line 2
	Scan Line 0 (1/2 Line)
BA52H	Vertical Pulses For The Even Field
	Scan Line 483 (1/2 Line)
	Scan Line 481
84E3H	Scan Line 3
84ABH	Scan Line 1
8000H	Vertical Pulses For The Odd Filed

Figure 30. The Format of The Digitized Video in The Ram

plus 77 sync pixels gives the following:

1. First scan line starts at 84ABH.
2. Applying the software window, the useful part of the first line starts at 84BDH.
3. Only the two-least significant bits of 84BDH contains useful information.
4. The useful part of the 1st line ends at 84DAH.
5. Only the seven most-significant bits of 84DAH contains useful information.
6. There are 28 bytes of useful information between the starting (84BDH) and ending points (84DAH).

The main analyzing routine has to know all the above information in order to analyze only the first scan line. This routine has to know the same type of information for the rest of the 399 lines (i.e. total # of lines to be analyzed is 400). This problem gets tougher because of the fact that all the odd scan lines reside in the other block of the memory and the analyzing routine has to switch between these two blocks as shown in Figure 24, p.46. This problem is solved by using a lookup table referred to as "table2." Table2 is a predefined table which has all the above information for 400 scan lines as shown in Figure 31. Table2 is made up by a separate program (not a part of system software residing in the EPROM). The output of this program is "table2" which is linked with the rest of system software and stored in the EPROM. The interrupt service routine then starts fetching the line addresses from beginning of the table and works its way down. The end of table is shown with 00H in the last

	One Byte	One Byte
Address: M	Starting Address of "Low Byte"	The Line 1 "High Byte"
M+2	# of Bits in First Byte	Ending Address of "Low Byte"
M+4	The Line 1 "High Byte"	# of Bits in Last Byte
M+6	# of Intermediate Bytes To be Processed	Starting Address of "Low Byte"
	The Line 2 "High Byte"	# of Bits in First Byte
	Ending Address of "Low Byte"	The Line 2 "High Byte"
	# of Bits in Last Byte	# of Intermediate Bytes To be Processed
	.	.
	.	.
	.	.
	Starting Address of "Low Byte"	The Line 400 "High Byte"
	# of Bits in First Byte	Ending Address of "Low Byte"
	The Line 400 "High Byte"	# of Bits in Last Byte
	# of Intermediate Bytes To be Processed	00H
	00H	00H
	00H	00H

Figure 31. The Format of The Table2

five locations. This will inform the main program when the analysis of the last line has been completed. Therefore, the main program doesn't have to be concerned with the numbers of lines to be analyzed. This will ease the job of changing the dimension of the software window by changing only the "table2."

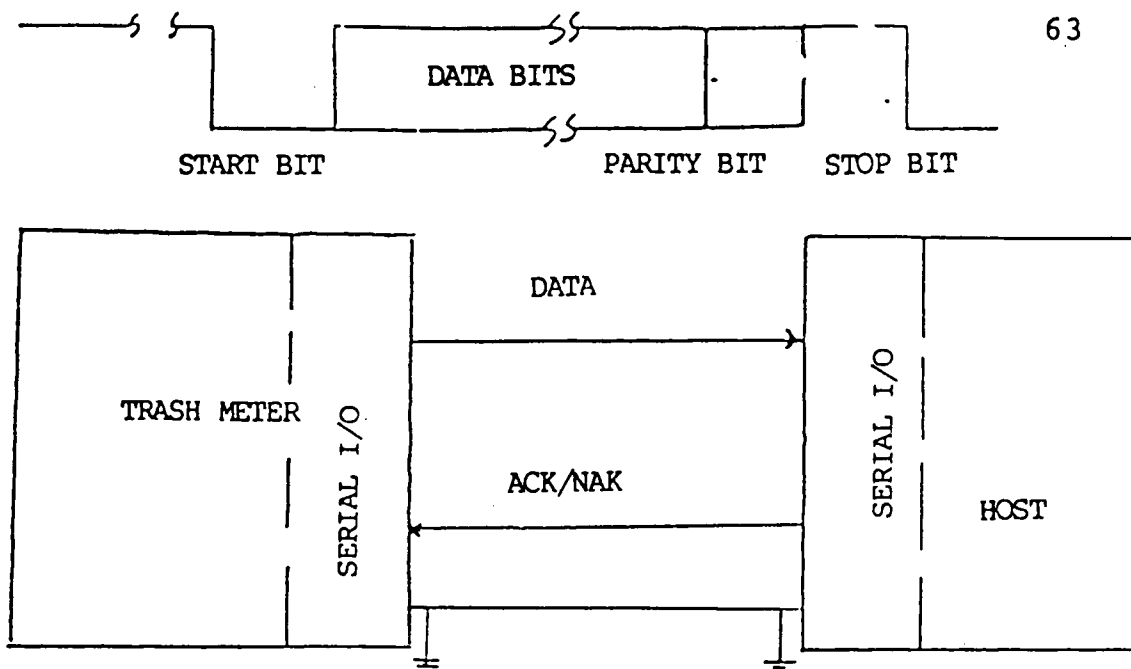
The interrupt service routine has to check each pixel to find out if it is a trash pixel or not. This is a very long process because there are 92,800 pixels (232 by 400) to be analyzed. In order to speed up the analysis, eight pixels (one byte) are read from the memory and analyzed together. This is done by using another lookup table referred as "TABLE1" as shown in Figure 32. Table1 is a predefined table which contains the summation of "1" present in an 8-bit number. For example the summation of "1" in 00H is 0 and summation of "1" in FFH and FEH is 8 and 7. Therefore the interrupt service routine fetches a byte of picture and computes the amount of trash pixels in that (trash pixel is "1") by simply looking up in the "TABLE1."

The communication format between the trash-meter and a host computer is shown in Figure 33. The trash-meter sends three pieces of the information to the host. The first two pieces are the trash area and count (each 4 digits long). The trash-meter software also performs a BCC (i.e. an exclusive-or on all data to be transmitted) and transmits the BCC after the trash count. The trash-meter then waits for ACK (acknowledge) or NAK (not acknowledge) for the host. The trash-meter retransmits the results in case of reception of a NAK.

One Byte

M	0
M+ 0000 0001	1
M+ 0000 0010	1
M+ 0000 0011	2
M+ 0000 0010	1
M+ 0000 0011	2
M+ 1111 1100	6
M+ 1111 1101	7
M+ 1111 1110	7
M+ 1111 1111	8

Figure 32. The Format of The Table1



BAUD RATE = 9,600

ASYNC. MODE

7-BIT DATA

EVEN PARITY

ONE STOP BIT

EXAMPLE

0011 0000 1

+

0011 0001 2

+

0011 0010 3

+

0011 0011 4

+

0011 0100 5

+

0011 0101 6

+

0011 0110 7

+

0011 0111 8

ASCII CODES

REPRESENTING TRASH AREA = 0123H

ASCII CODES

REPRESENTING TRASH COUNT = 4567H

0000 0000 9 > BCC PERFORMED ON ABOVE DATA

0100 0000 10 > END OF TRANSMISSION = 40H = @

TOTAL OF 10 ASCII CODES TRANSMITTED

Figure 33. The Serial Communication Format

B. Software Considerations

The entire system code is written completely in Z-80 assembly language. The codes are assembled and linked by the HP-64000 micro-processor development system. The final code is then programmed into an 8K EPROM (Intel 2764). The program is made of three parts which are:

1. The power-up routine
2. The interrupt service routine
3. The support routines

1. The Power-up Routine

The power-up program starts executing upon a power-up reset or a push-button reset and loads at 0000H. The flow chart of the power-up routine is shown in Figure 34. All the interrupts are disabled to make sure that the CPU goes through a proper initialization. The USART (8251A) is also initialized at the beginning which also serves as a software reset. The selection of the 8251A will cause a reset to all the flip-flops and counters(software reset). The interrupt mode one is set by the power-up routine. In this interrupt mode, the CPU will jump to location 0038H upon a maskable interrupt. However, the maskable interrupt must be enabled by the program before any interrupt may be serviced. After a complete initialization, the CPU waits for an interrupt occurrence. A DMA cycle may happen during the time which the CPU is waiting for an interrupt. As mentioned before, the DMA controller generates a maskable interrupt as soon as a picture frame is stored in the RAMs and the system bus is available for the CPU.

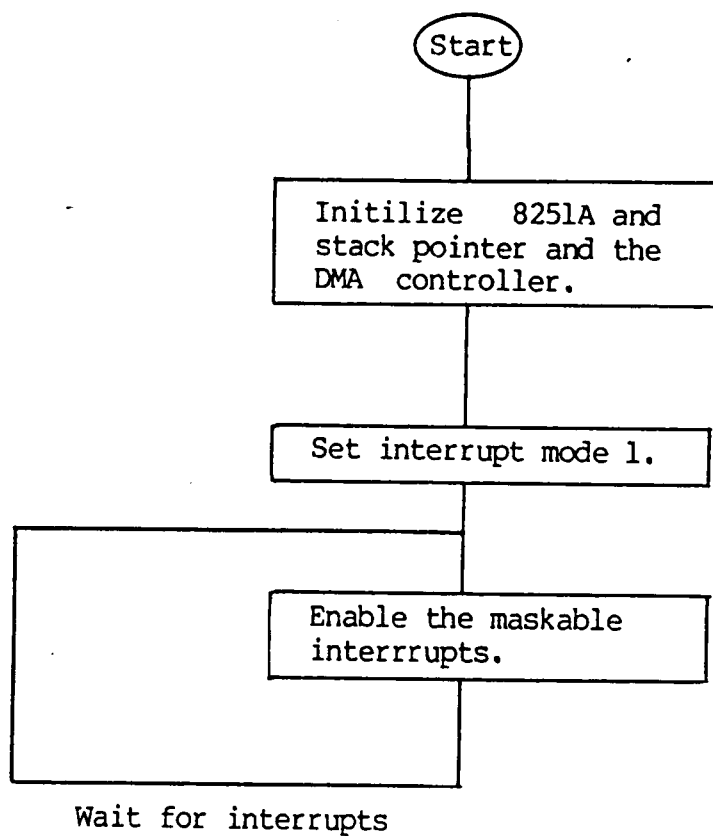


Figure 34. The Flow Chart of The Power-upe Routine

2. The Interrupt Service Routine

This is the routine which starts executing upon an occurrence of a maskable interrupt. The general flow chart of this routine is shown in Figure 35. The interrupt routine mainly consists of two parts. The first part analyzes the picture and computes the "trash-area" and the "trash-count." The second part transmits the results (i.e. trash-area and trash-count) to a host computer and takes care of all the communication problems such as error detections and retransmission. Both parts will be described in detail in the following paragraphs.

The flow chart of the first part is shown in Figure 36. There are some names written beside some of the blocks in the flowchart. These names are the name of macros used to accomplish what is inside that block. Macros are used versus subroutine primarily because of their speed. The "trash area" and the "trash count" are set to zero at the beginning. "TABLE2 POINTER" is a pointer which is pointing to the starting address of the TABLE2 minus seven. These are also two tables used to keep the starting, ending, and number of trash spots on a "reference" at a "data" line. These two tables are referred as TABLEA and TABLEB. "Pointer 1" is a pointer which points to either of these tables and shows where the ref start and ref end or data start and data end coordinates have to be stored. The pointer 1 points to the table1 at the beginning. The table pointer 2 is incremented by 7 so it will point to beginning of the table2. That is where all

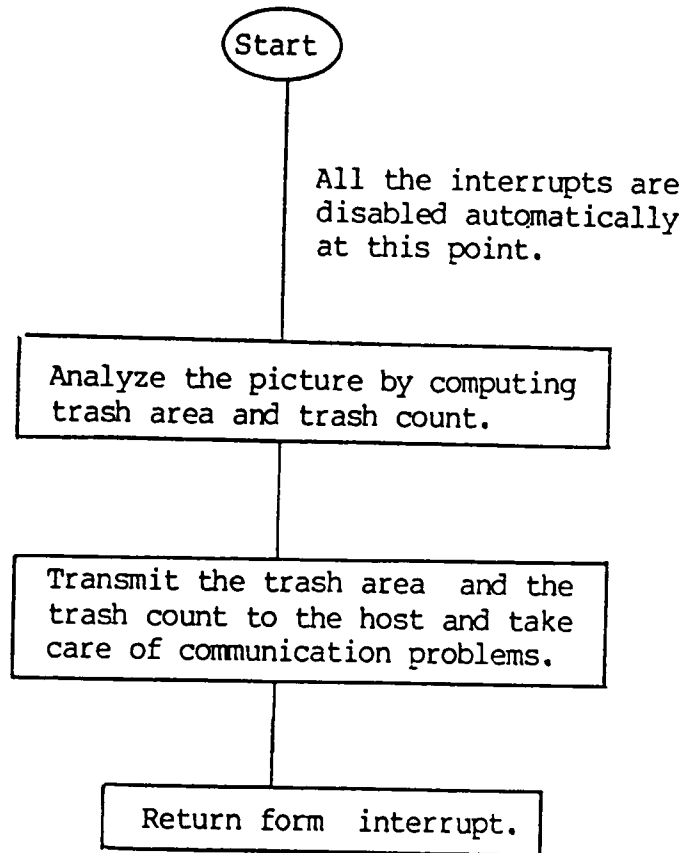


Figure 35. The General Flow Chart of The Interrupt Routine

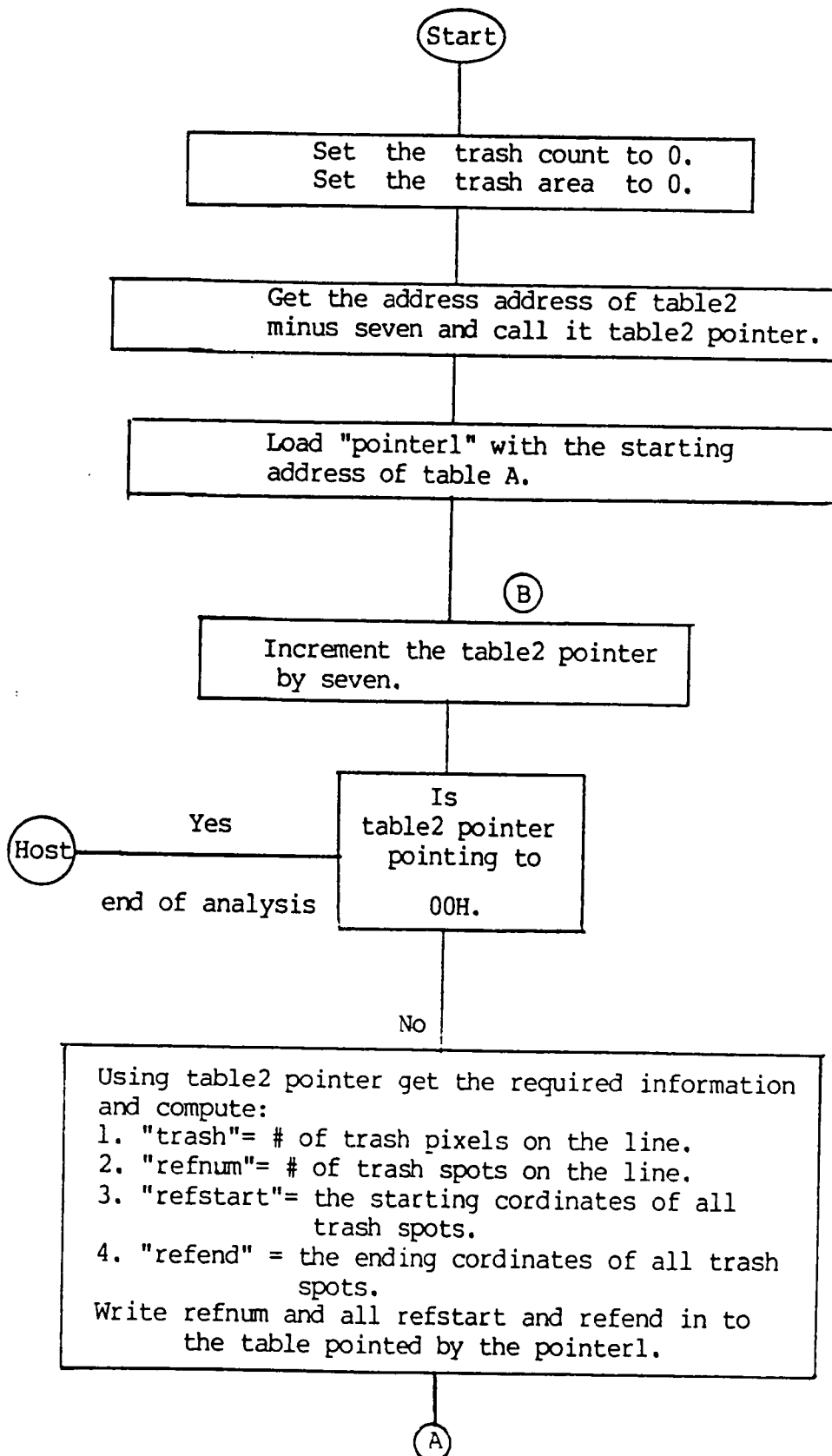


Figure 36. The Flow Chart of The First Part of Interrupt Routine

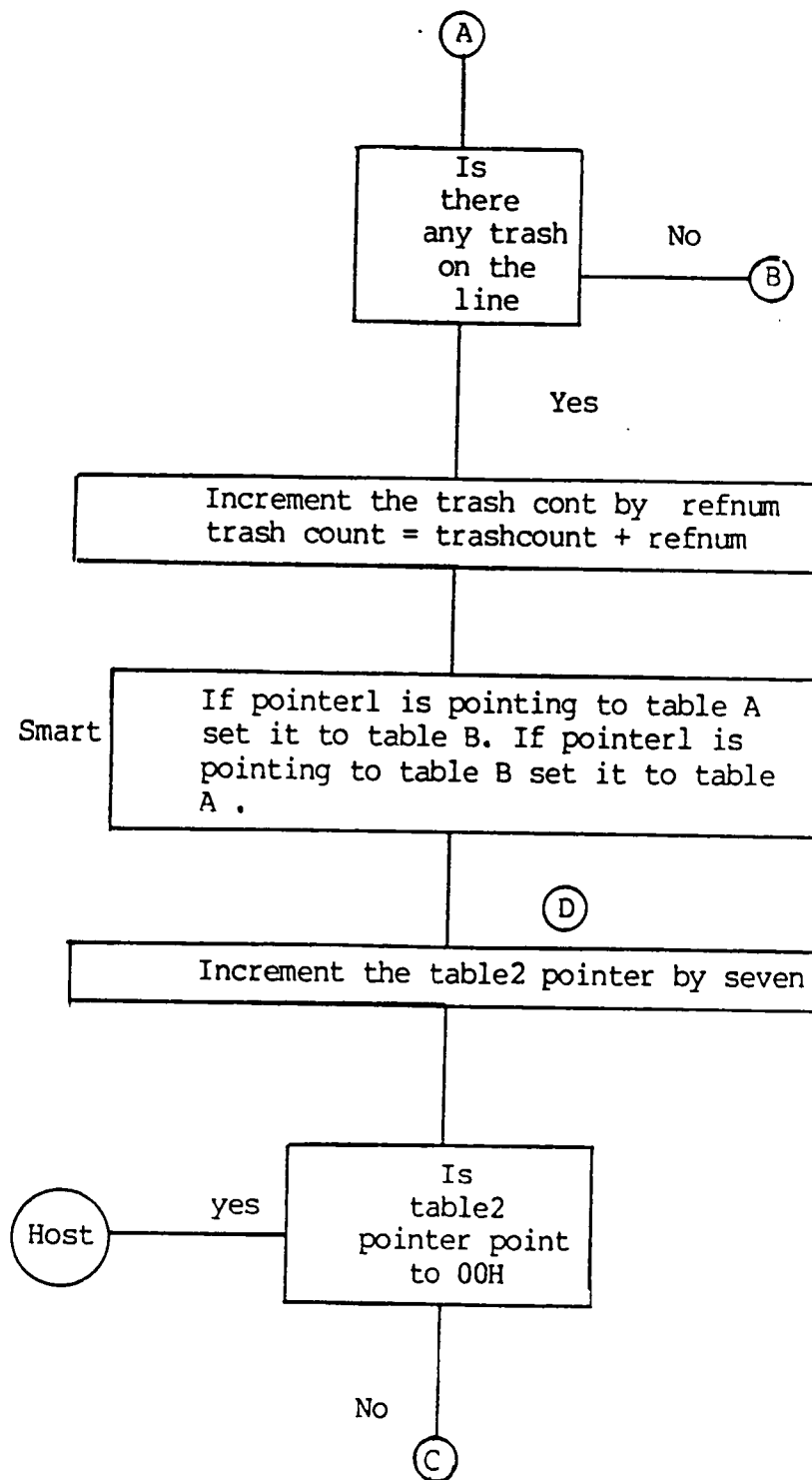


Figure 36. (Continued 1)

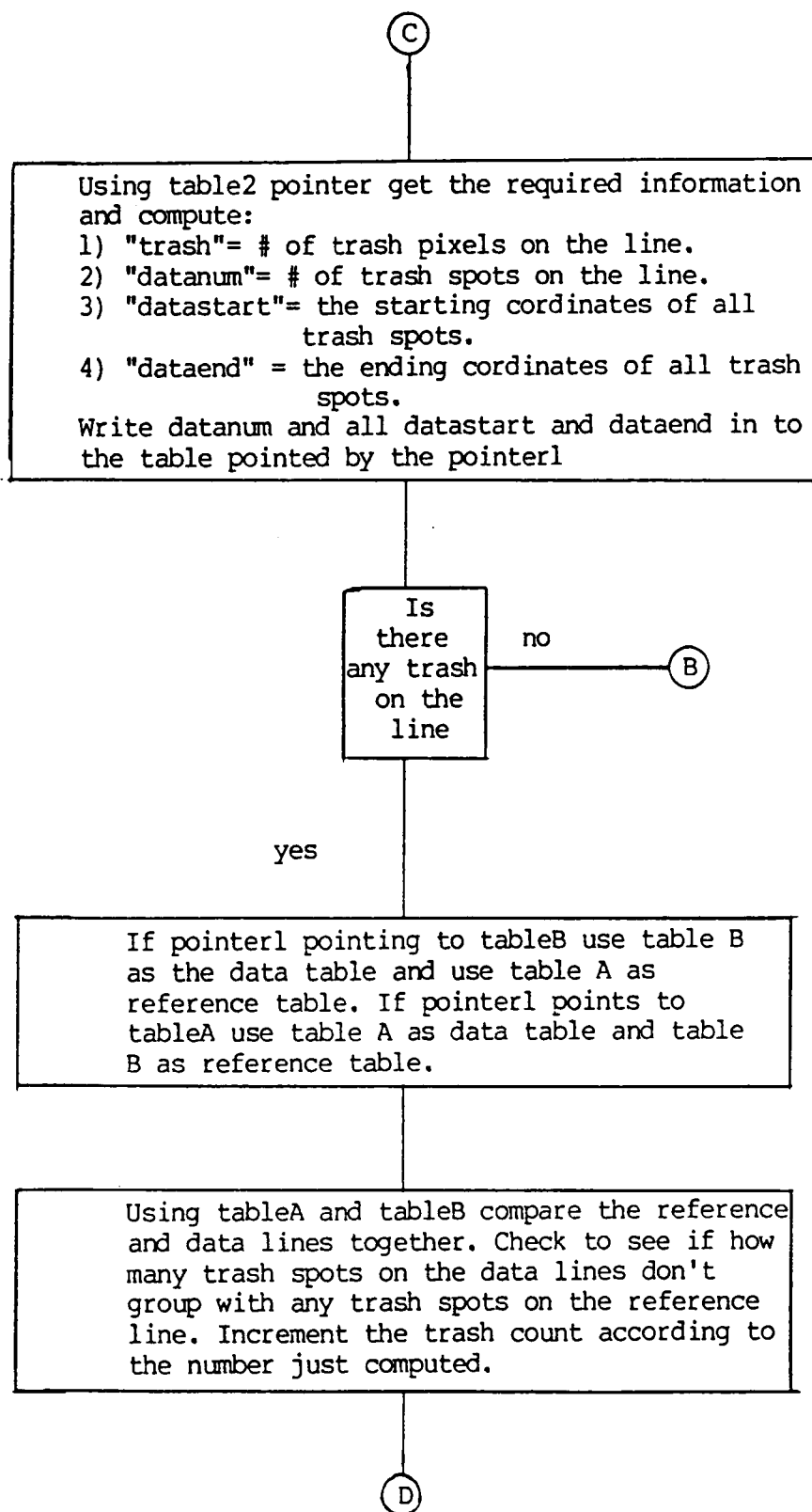


Figure 36. (Continued 2)

the required information for the first scan line are kept such as starting addresses and ending addresses of line one.

The high byte of the starting address for the next scan line (table pointer 1) is compared with 0 to check for the end of analysis. If it is zero, it is the end of analysis and the program control goes to the "host" block to transmit the results to the host computer. The next block is performed by a macro called "Nline." The "Nline" routine computes the number of trash pixels present on the line ("TRASH") and increment the trash area by using table1. The "Nline" also computes and writes the starting and ending coordinates of all trash spots present on the line in to the tableA. In the next block the program checks to see if there has been any trash pixels on the line which has just been analyzed. If not the program goes to point, otherwise it continues with the next block. The next block increments the trash count according to the number of trash spots observed in the last line.

The next block is a macro routine called "smart." This routine decides which table (A or B) has to be used for storing the datastart, dataend, and datanum. If tableB is assigned to be the table for data line table A will be assigned to reference line or vice versa. In the first pass through the smart routine, tableA has already been defined as the reference table, therefore tableB would be assigned as the data table. However, in the next pass through this routine, tableB keeps its content and will be reassigned as the new reference

line and the new information concerning the data line will be written in to tableA.

In the next two blocks, the table 2 pointer is incremented by seven, so a new data line can be fetched by using table 2. The end of table 2 is also checked. If the table 2 pointer is pointing to the end of table 2, the analysis will be terminated by jumping to "host" block.

The "Nline" routine is used again in the next block. The "Nline" will look at a new data line (using table 2 and table 2 pointer) and will store the data start, data end, and datanum in either table A or B which ever the pointer 1 points. It also upgrades the trash area if there is any trash present on the line.

The next three blocks are performed by a new macro called "compare." This macro first makes sure that the "data" line has some trash spots on it. After that, it will compare the "data" and the "reference" lines according to table A and B. If there are any trash spots on the "data" line which don't group with any spots on the "reference" line, the trash count is incremented according to the number of mismatch. Once the "compare" routine is finished, table 2 pointer is incremented and the process repeats again.

After finishing the analysis of the picture, the program control jumps to the "host" block. This is the second main part of the interrupt service routine which is shown in Figure 37. The trash area which is a 4-digit hex number, is broken up into 2-digit number, and their ASCII codes are computed. The same type of ASCII conversion is done on the trash count. The eight ASCII codes representing trash area

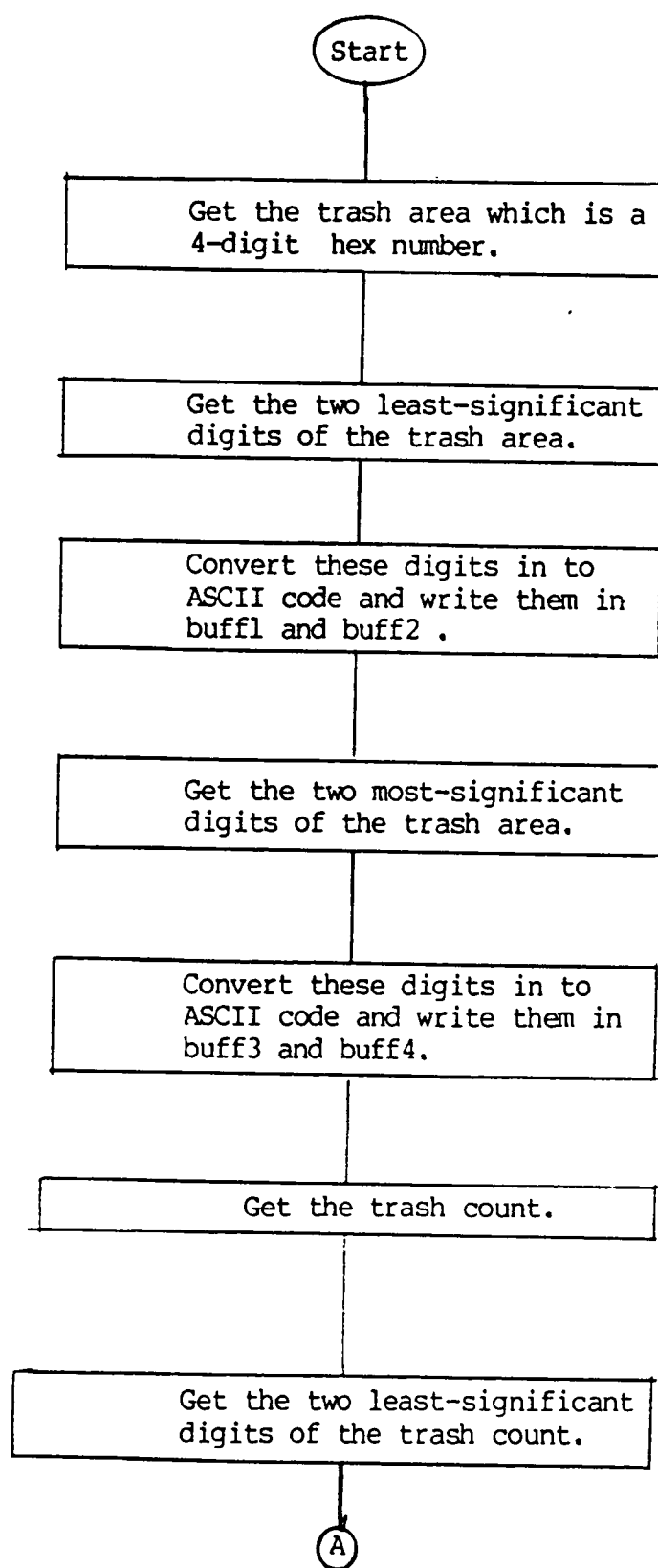


Figure 37. The Flow Chart of The Second Part of The Interrupt Routine

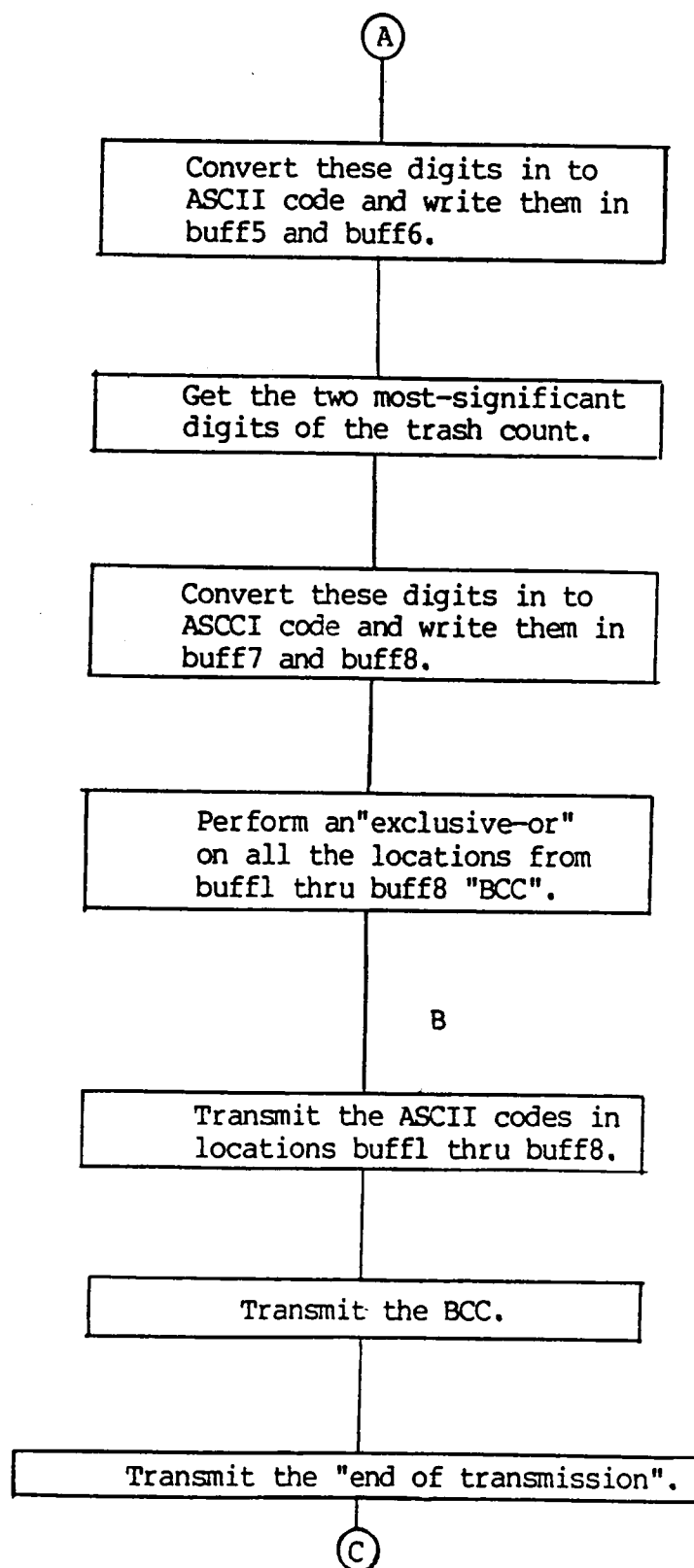


Figure 37. (Continued 1)

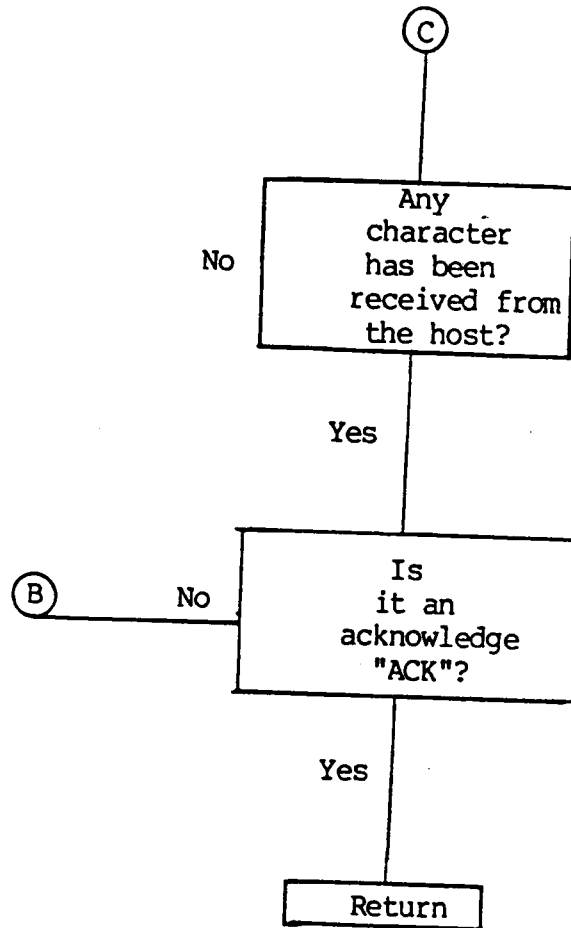


Figure 37. (Continued 2)

and count are exclusive-ored together and the result is called BCC.

The transmission format will be as follows:

1. Four ASCII codes for trash area
2. Four ASCII codes for trash count
3. One ASCII codes for BCC
4. One ASCII code for end of transmission character "@"

3. The Support Routines

The support routines are a combination of all the subroutines which get called and used by the power-up and the interrupt service routine. A list of all these routines may be found in appendix A.

CHAPTER III

PUSH-TO-TEST AND TROUBLE-SHOOTING

One of the powerful capabilities of a microprocessor-based instrument, is the ability to carry out a sequence of tests on itself. An ideal self-testing would be the capability of testing all these four block thoroughly upon power-up, push-to-test, and periodically during normal instrument operation. It would also have to be able to report the malfunctioning device to the user. Having this capability provides not only a more confident working instrument but also easier trouble shooting and repair procedure.

Unfortunately, the author has not been able to impliment such an ideal self-testing because of lack of time and room on the board. However, a push-to-test feature is implemented in the trash-meter. As Figure 1, p.3, shows, there are four main blocks involved in the trash-meter circitry:

1. The CPU Block
2. The Memory Block
3. The Start Block
4. The DMA Controller Block

The CPU block and at least part of ROM must be operational in order to run a push-to-test. Upon a push-to-test, the CPU runs a diagnostic program (residing in the ROM) which will test the RAMs, ROM, and the USART. If all the tests are passed, a LED will light up to inform the operator with the test result. The flow chart of the diagnostic program is shown in Figure 38. The USART block is exercised first. The CPU

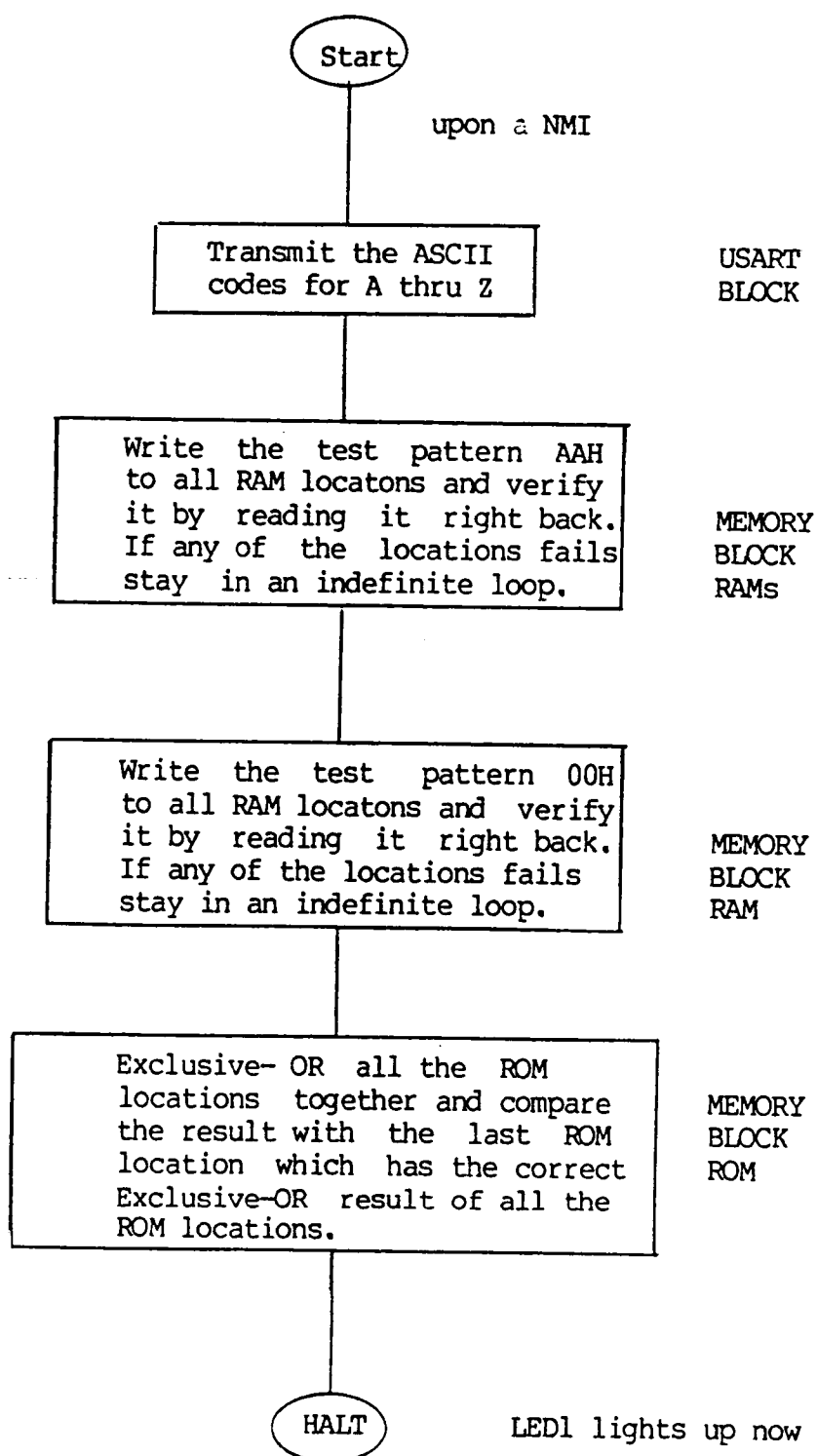


Figure 38. The Flow Chart of The Push-to-test routine

transmits the A thru Z characters which could be displayed in the host monitor with proper communication programming. The memory block consists of RAMs and ROM which require two different test algorithms. The RAMS are tested by writing a test pattern to all the locations and read them back for verification. AAH and 55H are the test patterns which are used. They exercise all the bits to their "low" and "high" states. If any of the RAM locations fails the test, the program will stay in an infinite loop, therefore, the output LED will not light up. The ROM is tested by exclusive-oring all the the ROM locations together and compare the result with the last ROM location. If they do not match, the program stays in an infinite loop again. This, of course, requires that the last ROM location must have been programmed. Due to the complexity of the DMA controller block, a simple test routine is not feasible without extra hardware. If all the above tests pass, the CPU then lights up the LED1.

The switch (SW1) shown in Figure 3, p.9, initiates the push-to-test. Pushing this switch will cause a "high" to "low" transition on the non-mask interrupt input (NMI) of the Z-80A CPU. The diagnostic routine is the routine which the CPU starts executing upon a NMI. In case of a NMI, the CPU will fetch the next instruction placed at 0066H which transfer the program control to the diagnostic program. If all the test routine pass, the program then starts executing a HALT instruction. This brings the HALT pin of the CPU "low" which in turn will light up the LED1. After a push-to-test, a push-button reset must be applied in order to get the system back in to the normal operation.

The advantages of the push-to-test just discussed are its simplicity and the hardware requirements (i.e. only a push-button switch and a LED). However, the disadvantage is its limited test coverages.

CHAPTER IV

THE TRASH-METER EVALUATION

This chapter describes the operational response characteristics of the trash-meter. Unfortunately, there is not any standard procedure for the trash-meter evaluation at this point in time. Area linearity, particle size, and particle-size sensitivity are three parameters for which the trash-meter is evaluated. Various calibration papers, each having different dot sizes and total dot area are used to evaluate the above parameters. The test apparatus is set up as shown in figure 39. The camera is set in such a way that only the observation window is covered by the camera for best picture resolution (i.e. maximum number of the pixels inside the observation window).

A. Particle-Size Sensitivity

The smallest trash size which may be detected by the trash-meter is a function of the followings:

1. The actual size of each pixel in the camera sensor unit.
2. The distance between the camera and the object being observed.
3. The threshold-setting which was mentioned in chapter 2.

With the above test set-up, the first two factors are not variable anymore. Therefore the only variable factor which may change the particle-size sensitivity is the threshold setting. In order to compute the actual size of a picture pixel (not the same as the pixels in the camera sensor unit), a one-inch by

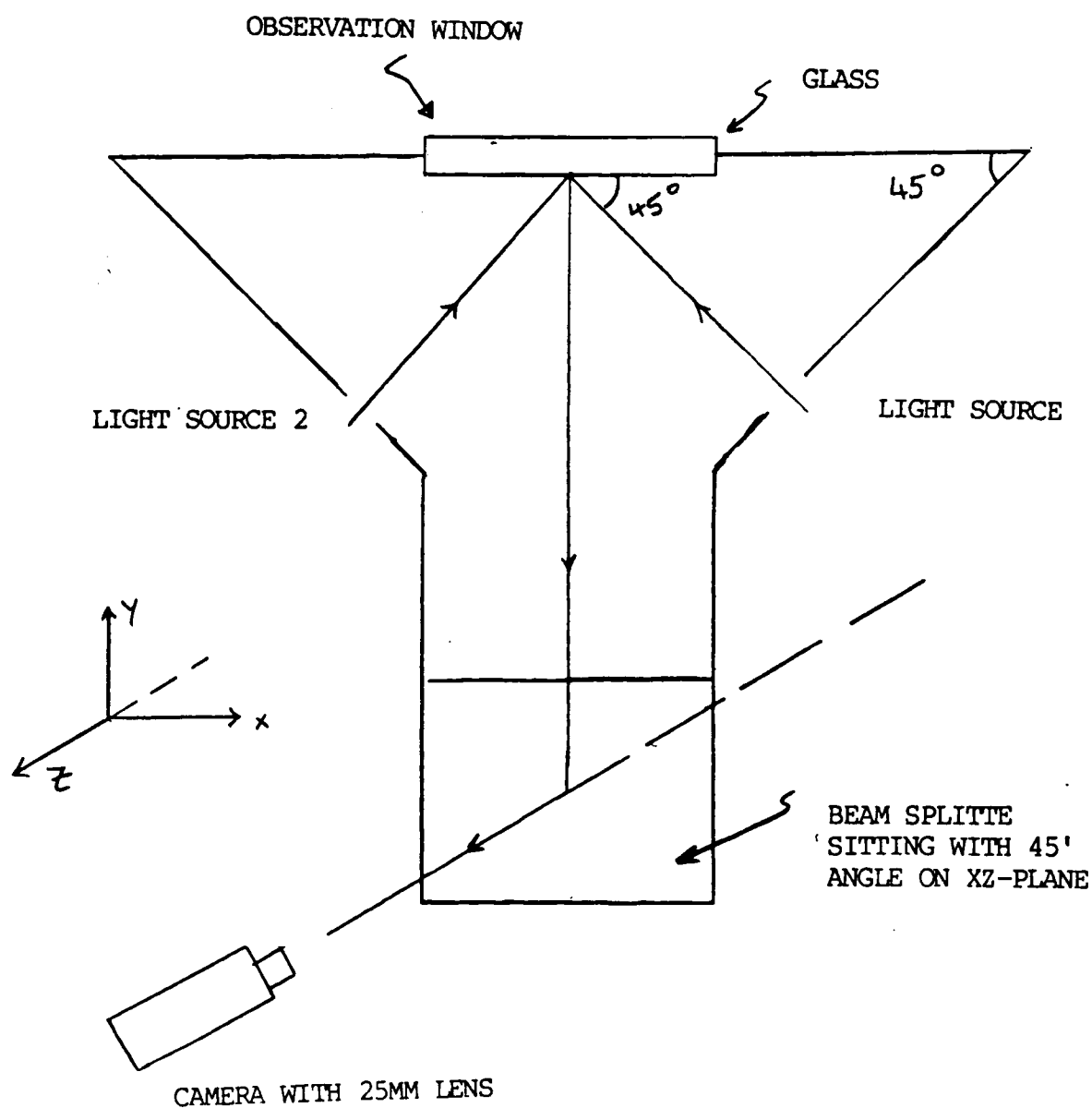


Figure 39. The Test Setup

one-inch solid black square is placed on the observation window. The total number of black pixels are measured for different threshold setting. Five readings are taken for a given threshold-setting, and the results are averaged. The results are plotted for different threshold-voltages as shown in Figure 40. The voltage comparator (U34) shown in Figure 19, p.35, explains this curve. The comparator compares the video voltage level to a D.C. level (threshold-level). Any video voltage level less than the threshold-level is considered black. Due to the lack of complete light uniformity on the observation window, some of the spots are darker than the others. Therefore, these spots tend to increase the trash count as the threshold-setting increases. This phenomena is quite obvious when the threshold-voltage increases from zero volts to 0.21V. The curve stays flat for the threshold-voltages greater than 0.21V. This is because all the black spots on the picture have a lower voltage-level (as a part of video signal) than the threshold-voltage of 0.21V. Consequently, they are all seen as black spots by the trash-meter.

Threshold setting of 0.21V yields approximately 11,298 pixels per square inch. Therefore, each picture pixel has an area of approximately 8.85×10^{-5} square inches. This is the smallest particle which may be detected by the trash-meter with a threshold-setting of 0.21V or above.

B. Area Linearity And Stability

Four different calibration papers are used to test the area linearity. These calibration papers are shown in Figure 41a through 41d. All the circles are identical to each other and are 0.2 ± 0.002

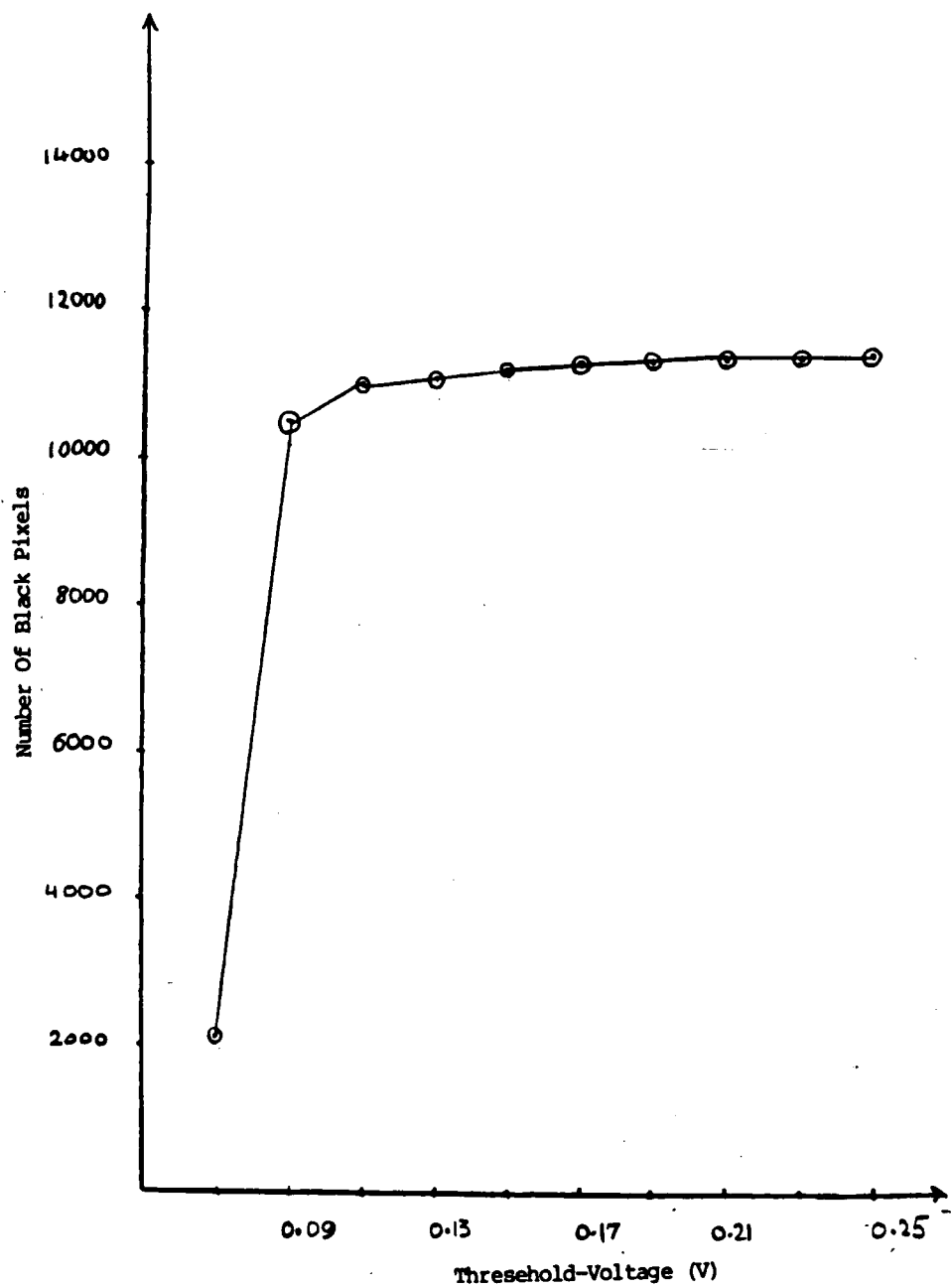
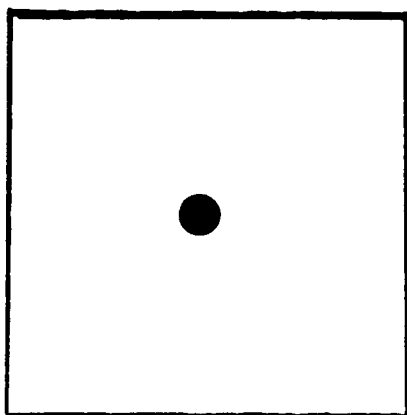
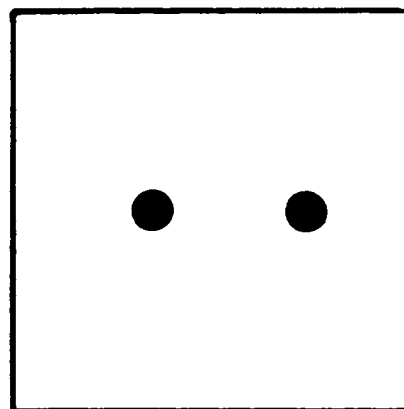


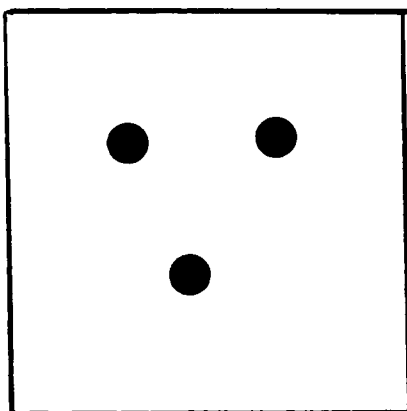
Figure 40. The Particle-sensitivity graph



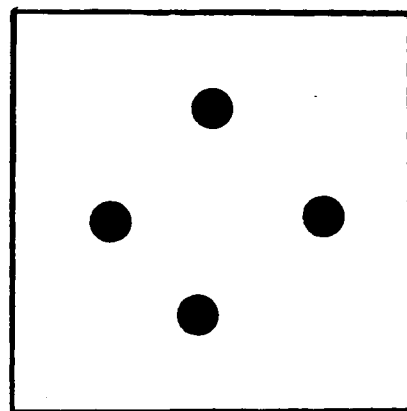
(a)



(b)



(c)



(d)

Figure 41. The Calibration Papers

inches in diameter. Ten readings are taken on each calibration paper and results are are averaged. The results (the number of black pixels present in each calibration paper) are shown in Table 1. The graph in Figure 42 shows the area linearity of the trash-meter. The non-linearity error may be computed as follow:

$$\text{area of paper 42b/area of paper 42a} = 728/363 = 2.006$$

$$\text{non-linearity error} = 2.006 - 2 * 100 / 2 = 0.3\%$$

$$\text{area of paper 42c/area of paper 42a} = 1102/363 = 3.036$$

$$\text{non-linearity error} = 3.036 - 3 * 100 / 3 = 1.2\%$$

$$\text{area of paper 42d/area of paper 42a} = 1441/363 = 3.970$$

$$\text{non-linearity error} = 3.970 - 4 * 100 / 4 = 0.75\%$$

So the maximum non-linearity error is 1.2% which may be caused by the tolerance of the spots in the calibration papers.

C. Summary And Conclusion

The development of the trash-meter was based on the requirement of the Spinlab-900 high-volume testing instrument. The trash-meter is capable of analyzing 92,800 pixels through a 3.5" by 3.5" window in less than one second. The video-scanning trash-meter can do a reasonably accurate job of determining trash area and trash count.

The trash-meter software may be upgraded by adding a "histogram of different trash sizes" to it. By defining the different trash sizes by the user, the trash-meter can compute the number of trash spots belong in a different trash-size category.

The trash-meter may also be used as a programmable general

Table 1. The Area-Linearity Table

RUN #	ONE SPOT	TWO SPOTS	THREE SPOTS	FOUR SPOTS
1	371	732	1112	1454
2	368	735	1092	1417
3	356	735	1097	1426
4	364	734	1090	1452
5	354	727	1103	1448
6	355	726	1108	1423
7	365	727	1103	1418
8	368	724	1108	1419
9	367	725	1106	1452
10	364	722	1102	1454
AVERAGE	363	728	1102	1441
STANDARD DEVIATION	5.7	4.6	7.3	16.6

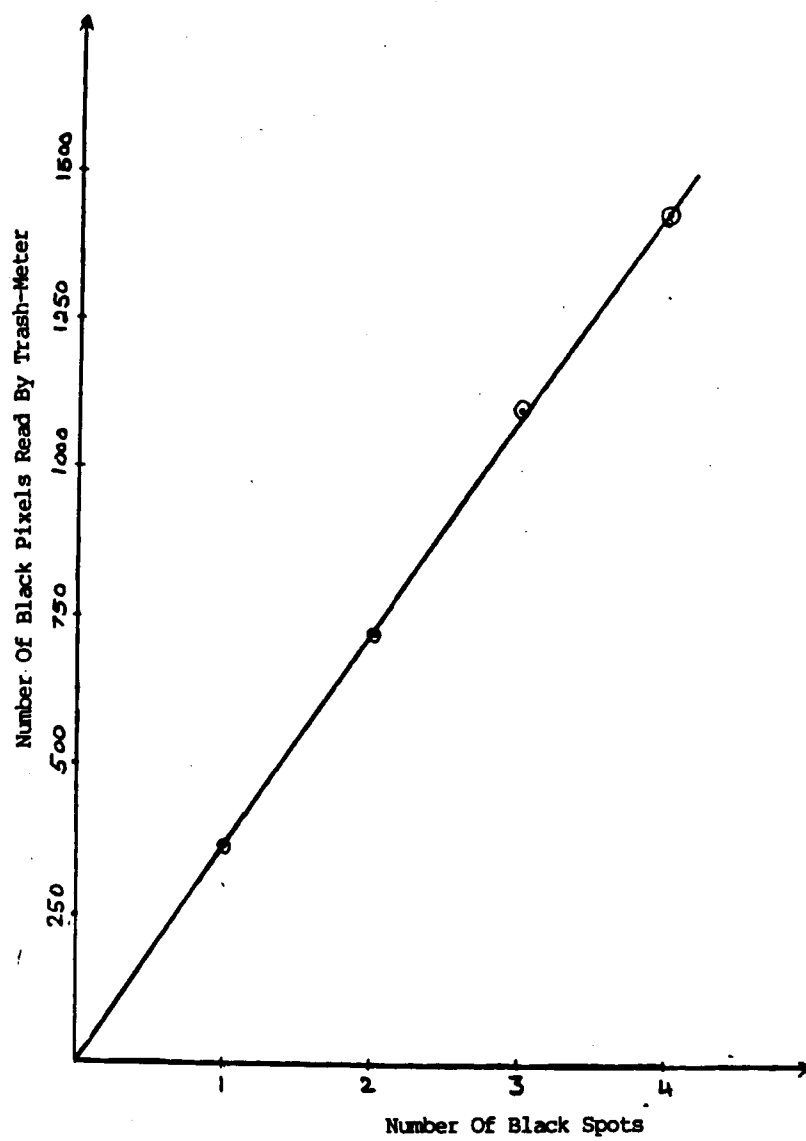


Figure 42. The Area-Linearity graph

purpose image analyzer by just changing the trash-meter software. In this case, the image analyzer may be programmed through its serial I/O port.

LIST OF REFERENCES

LIST OF REFERENCES

- Vision For Automation CCD3000 and 4000, Fairchild Company, 1982.
- Intel Microsystem Components Handbook Volume 2, Intel Corporation, 1984.
- Intel Memory Components Handbook, Intel Corporation, 1983.
- I C Memories Databook, Hitachi America, LTD, 1982.
- The TTL Databook For Design Engineer, Second Edition, Texas Instrument Inc., 1976.
- John B. Peatman, "Microcomputer-Based Design," McGraw Hill, Inc., 1977.
- Linear Division Products, Fairchild Company, 1982.
- William Barden, Jr., "The Z-80 Microcomputer Handbook," Howard W. Sons And Co., INC, 1983.
- Robert A. Taylor, "Using High Speed Image Analysis To Estimate Trash In Cotton," USDA Cotton Quality Research Station, Clemson, S.C., 1982.

APPENDIXES

APPENDIX A

PROGRAM LISTING OF THE SYSTEM

THE INTERRUPT SERVICE ROUTINE

```

"Z80" LIST XREF
TABLE2      ORG      0300H
            EQU      0FF9H      ;STARTING ADDRESS FOR
                                ;LOOKUP TABLE2 MINUS
                                ; SEVEN.
TABLE1      EQU      0070H      ;STARTING ADDRESS FOR
                                ;LOOKUP TABLE1.
TEMP1       EQU      0FFFEH
TEMP2       EQU      0FFFFH
TEMP3       EQU      0FFF6H
SAVE        EQU      0FFFCH
REFNUM      EQU      0F4FFH      ;NUMBER OF TRASH SPOTS
                                ;ON THE REFERENCE LINE.
REF         EQU      0F500H      ;STARTING ADDRESS FOR
                                ;TABLE WHICH KEEPS THE
                                ;STARTING AND ENDING
                                ;CORDINATES OF
                                ;REFERENCE TRASHES.
DATANUM     EQU      0F9FFH      ;NUMBER OF TRASH SPOTS
                                ;ON THE DATA LINE.
DATA        EQU      0FA00H      ;STARTING ADDRESS FOR
                                ;DATA TABLE.
FLAG        EQU      0FFFAH      ;THIS FLAG IS SET IF
                                ;A TRASH SPOT IS
                                ;DETECTED AND IS UNDER
                                ;STUDY.
XPOINTER    EQU      0FFF8H      ;A POINTER IN THE X-
                                ;DIRECTION WHICH KEEPS
                                ;THE X-CORDINATE OF
                                ;THE PIXEL UNDER
                                ;STUDY
SPOT        EQU      0FFF4H      ;HOLDS THE TRASH COUNT
                                ;"IX"REGISTER HOLDS
                                ;THE TOTAL TRASH AREA.
            EXT      HOST
;
;
; THIS MACRO WILL ACCUMLATE THE TRASH AREA IN IX REGISTER
;
;
INCREMENT   MACRO
            LD      A,B          ; [B+TABLE1]——>C
            LD      [TEMP1],A
            LD      [TEMP3],IY   ;SAVE IY
            LD      IY,[TEMP1]
            LD      C,[IY+TABLE1]
            LD      IY,[TEMP3]   ;GET BACK IY
            XOR     A
            LD      B,A          ;IX+BC——>IX
            ADD     IX,BC        ;ACCUMLATE TRASH
            MEND                . RE^
;

```

"Z80" LIST XREF

	ORG	0300H	
TABLE2	EQU	0FF9H	;STARTING ADDRESS FOR ;LOOKUP TABLE2 MINUS ; SEVEN.
TABLE1	EQU	0070H	;STARTING ADDRESS FOR ;LOOKUP TABLE1.
TEMP1	EQU	0FFFEH	
TEMP2	EQU	0FFFFH	
TEMP3	EQU	0FFF6H	
SAVE	EQU	0FFFCH	
REFNUM	EQU	0F4FFH	;NUMBER OF TRASH SPOTS ;ON THE REFERENCE LINE.
REF	EQU	0F500H	;STARTING ADDRESS FOR ;TABLE WHICH KEEPS THE ;STARTING AND ENDING ;CORDINATES OF ;REFERENCE TRASHES.
DATANUM	EQU	0F9FFH	;NUMBER OF TRASH SPOTS ;ON THE DATA LINE.
DATA	EQU	0FA00H	;STARTING ADDRESS FOR ;DATA TABLE.
FLAG	EQU	0FFFAH	;THIS FLAG IS SET IF ;A TRASH SPOT IS ;DETECTED AND IS UNDER ;STUDY.
XPOINTER	EQU	0FFF8H	;A POINTER IN THE X- ;DIRECTION WHICH KEEPS ;THE X-CORDINATE OF ;THE PIXEL UNDER ;STUDY
SPOT	EQU	0FFF4H	;HOLDS THE TRASH COUNT ;"IX"REGISTER HOLDS ;THE TOTAL TRASH AREA.

EXT HOST

```

;
;
; THIS MACRO WILL ACCUMLATE THE TRASH AREA IN IX REGISTER
;
;

```

```

INCREMENT      MACRO
                LD      A,B                ; [B+TABLE1] —>C
                LD      [TEMP1],A
                LD      [TEMP3],IY         ;SAVE IY
                LD      IY,[TEMP1]
                LD      C,[IY+TABLE1]
                LD      IY,[TEMP3]         ;GET BACK IY
                XOR     A
                LD      B,A                ;IX+BC——>IX
                ADD     IX,BC               ;ACCUMLATE TRASH
                MEND                                ;AREA

```

```

;
;
;THIS MACRO INCREMENT THE X AXIS POINTER BY 8
;
;
UPGRADE          MACRO
                  LD      A,[XPOINTER]      ;INCREMENT THE X
                  ADD     A,8                ; POINTER BY 8
                  LD      [XPOINTER],A      ;AND SAVE IT.
                  XOR     A
                  MEND

;
;
;THIS MACRO WILL FIND THE STARTING PIXEL FOR THE TRASH ON THE LINE
;AND SAVE IT.
;
;
FINDSTART        MACRO
                  EXX
                  INC     B                ;B HAS THE # OF
                                           ;TRASHES ON THE LINE
                  LD      A,[XPOINTER]
                  LD      [HL],A          ;SAVE THE "START"
                  INC     HL              ;INCREMENT FOR NEXT
                                           ;TIME
                  LD      A,B             ;SET THE FLAG BY ANY
                  LD      [FLAG],A        ;VALUE BUT 0
                  EXX
                  XOR     A
                  MEND

;
;
;THIS MACRO WILL FIND THE END PIXEL OF THE TRASH AND SAVE IT.
;
;
FINDEND          MACRO
                  EXX
                  LD      A,[XPOINTER]
                  LD      [HL],A
                  INC     HL
                  EXX
                  XOR     A
                  LD      [FLAG],A        ;RESET FLAG TO SHOW
                                           ;THE END OF A TRASH
                  MEND

;
;
;
;
;
;THIS MACRO FINDS THE STARTIND AND ENDING

```

```

;      COORDINATES OF A TRASH ON EACH LINE AND THE NUMBER OF TRASH
;      PIXELS AND TRASH SPOTS AVAILABLE ON THE LINE AND STORE THEM
;      ON A TABLE WHERE IS DEFINED BY THE CALLING PROGRAM.
;
;
NLINE      MACRO      &ST1,&PTT2,&NEXT1,&NN
LD          D,[IY+6]      ;D CONTAINS THE # OF
                        ;BYTES TO BE PROCESSED

XOR        A
LD          [XPOINTER],A
LD          L,[IY+3]      ;LOW BYTE OF ADD. FOR
                        ;END BYTE

LD          H,[IY+4]
LD          A,[IY+5]      ;MASK FOR THE END PART
AND         [HL]
LD          [HL],A        ;RESTORE IT IN MEME.
LD          L,[IY+0]      ;DO THE SAME FOR START
LD          H,[IY+1]
LD          A,[IY+2]
AND         [HL]
LD          B,A
&ST1:      XOR          A
CP          B              ;IS IT ZERO
JP          NZ,&PTT2        ;IF NOT ZERO GO TO PT2
LD          A,[FLAG]      ;CHECK THE FLAG
LD          B,A
XOR          A
CP          B
JP          Z,&NEXT1        ;NO TRASH IN PROGRESS
;
;
                        ;TRASH JUST FINISHED
                        ;FIND THE END AND SAVE
                        ;IT
JP          &NEXT1        ;GO TO TRY THE NEXT
                        ;BYTE

;
;TRASH IN PROGRESS
;
&PTT2:      INCREMENT      ;IT WAS A TRASH SO
                        ;ACCUM.THE TRASH AREA
LD          A,[FLAG]      ;CHECK THE FLAG
LD          B,A
XOR          A
CP          B
JP          NZ,&NEXT1        ;IF A NEW TRASH GO
                        ;TO "NEXT"
                        ;FIND START
                        ;SAVE IT
&NEXT1:      UPGRADE      ;XPOINTER=XPOINTER+8
INC          HL            ;GET READY FOR NEXT
LD          B,[HL]        ;PICK THE NEXT BYTE
DEC          D             ;DECREMENT THE BYTE

```

```

;COUNTER
;LINE FINSHED YET?
JP      NZ,&ST1
LD      A,[FLAG]
LD      B,A
XOR     A
CP      B
JP      Z,&NN
FINDEND
&NN:    EXX
LD      A,B
LD      [DE],A
EXX
MEND

;SAVE THE NUMBER OF
;TRASHES ON THE LINE
;THE TRASH AREA

;
;
;
;THIS MACRO DIFFRENTIATE BETWEEN RE. AND DATA LINES BY SETTING
;HL DE AND C REGISTERS IN ALT. SET. IT ALSO CHECKS FOR THE END OF
;THE PICTURE AND RETURN TO THE MAIN PROGRAM AT END.
;
;
SMART    MACRO
EXX
XOR     A
CP      C
JP      Z,H1
;
;SET FOR DATA THEN
;
LD      C,A
LD      HL,DATA
LD      DE,DATANUM
LD      B,A
EXX
JP      C11
;
; SET FOR REF
;
H1:      EXX
LD      B,A
INC     C
LD      HL,REF
LD      DE,REFNUM
EXX
C11:     NOP
;1----->C

```

```

MEND
;
;
;
;THIS ROUTINE WILL COMPARE TWO LINES TOGETHER AND DECIDES IF TRASH
;SPOTS ON THESE LINES ARE OVERLAPPING OR NOT.
;
;
COMPARE      MACRO
              EXX
              XOR      A
              CP        C           ;WHICH TABLE IS REF
                                      ;OR DATA
              EXX
              JP        NZ,T1
;
              LD        HL,REFNUM
              LD        DE,DATANUM
              JP        T2           ;START COMPARING
;
T1:           LD        HL,DATANUM
              LD        DE,REFNUM
;
;COMPARISON STARTS HERE
;
T2:           LD        A,[DE]       ;B HAS THE # OF TRASH
              LD        B,A         ;ON THE DATA LINE
              XOR      A           ;IS # OF TRASH SPOTS
              CP        B           ;ON DATA LINE EQUAL
              JP        Z,HERE1     ;ZERO
              LD        C,[HL]      ;C HAS # OF TRASH ON
                                      ;REF. LINE.
              LD        [SAVE],HL   ;SAVE # OF TRASH ON
                                      ;REFERENCE
              INC        DE         ;POINT TO TABLE
              INC        HL         ;POINT TO TABLE
LOOP:         LD        A,[DE]      ;A=ST.DATA(I)
              INC        HL         ;[HL]=END REF(J)
              CP        [HL]
              JP        Z,T5        ;IF ADJACENT GO NEXT
              JP        P,T3        ;IF STDATA>END REF
                                      ;TRY NEXT REF
              INC        DE
              LD        A,[DE]      ;A=END DATA(I)
              DEC        HL         ;[HL]=ST. REF(J)
              CP        [HL]
              JP        Z,T5        ;IF ADJACENT GO NEXT
              JP        M,T4        ;IF END DATA(I)<
                                      ;ST. REF(J) THEN IT
                                      ;IS ANEW TRASH
              JP        T5          ;TRY NEXT DATA BECAUSE

```

```

;TRASHES ON BOTH LINE
;ARE ADJACENT
;LAST TRASH ON REF LINE
T3:      DEC      C
        XOR      A
        CP       C
        JP       Z,T4      ;IF IT IS THEN THERE
                                ;IS A NEW TRASH
                                ;GET READY FOR NEXT
                                ;REF TRASH
        INC      HL
        JP       LOOP

;
;IT WAS A NEW TRASH SO INCREMENT THE TRASH SPOT COUNT
;
T4:      LD       [TEMP3],IY      ;SAVE IY
        LD       IY,[SPOT]      ;INCREMENT THE
        INC      IY              ;NUMBER OF TRASH
        LD       [SPOT],IY      ;SPOTS
        LD       IY,[TEMP3]

;
;
T5:      DEC      B              ;ANY MORE TRASH ON
        XOR      A
        CP       B
        JP       Z,T6      ;IF NOT RETURN
        INC      DE          ;GET READY FOR NEXT
                                ;TRASH ON THE DATA
                                ;LINE
        LD       HL,[SAVE]      ;C HAS THE # OF TRASHES
        LD       C,[HL]        ;ON REF LINE
        INC      HL
        JP       LOOP
T6:      NOP
        MEND

```

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
; MAIN INTERRUPT ROUTINE STARTS HERE
;
;
;SET THE FIRST LINE AS REFERENCE LINE BY SETTING HL AND DE
;
        XOR      A
        LD       IX,00          ;RESET TRASH AREA
        LD       [SPOT],IX      ;RESET TRASH COUNT
        LD       IY,TABLE2      ;GET TABLE2
        LD       [TEMP2],A      ;RESET [TEMP2]
HERE1:   EXX
HERE2:   LD       C,1           ;SET THE REF LINE
        LD       HL,REF        ;SHOWS THE REF.
                                ;STARTING ADDRESS OF
                                ;REF. TABLE.

```



```

;
;LD          BC,7          ;CHECK FOR END OF
;ADD         IY,BC        ;FRAME
;XOR         A
;CP          [IY+1]
;JP          Z,FINISH      ;IF FINISHED GO TO
;                                     ;"FINISH".
;
;
;NLINE          STT,PTT,NEE,NN2 ;FIND THE NUMBER OF
;                                     ;TRASH SPOTS , THIER
;                                     ;STARTING , ENDING
;                                     ;CORDINATES AND TOTAL
;                                     ;NUMBER OF TRASH
;                                     ;AND STORE THEM ON A
;                                     ;TABLE.
;
;COMPARE
;                                     ;COMPARE TWO ADJACENT
;                                     ;LINES TOGETHER.
;
;CHECK FOR THE END OF PICTURE AND INCREMENT POINTER FOR TABLE2
;
FINISH:         JP          HERE4      ;GET THE NEXT LINE
CALL           HOST                ;TX THE RESULT TO HOST
RET
;PIXELS.

```

THE SUPPORT ROUTINE


```

;
;           LD           A,[SPOT]           ;GET THE NUMBER OF
;           LD           C,A               ;TRASH SPOTS.
;           CALL         BXASH              ;CONVERT THE LOWEST
;                                           ;TWO BITS TO ASCCI
;                                           ;AND PLACE THEM IN
;                                           ;BUFF5 AND BUFF6.
;
;           LD           A,[SPOT2]          ;CONVERT THE HIGHEST
;           LD           C,A               ;TWO BITS OF TRASH
;           CALL         BXASH              ;AREA IN TO ASCII
;                                           ;AND PLACE THEM IN
;                                           ;BUFF7 AND BUFF8.
;
;
;           CALL         INIT              ;INITILIZE 8251
;
;
;TRANSMIT THE TRASH AREA AND TRASH COUNT ASCCI CODES
;
;
KEEPON:     LD           A,[BUFF3]
;           LD           B,A
;           CALL         TX
;
;           LD           A,[BUFF4]
;           LD           B,A
;           CALL         TX
;
;           LD           B,A
;
;           CALL         TX
;           LD           A,[BUFF2]
;           LD           B,A
;           CALL         TX
;
;;
; TX THE COUNT
;           LD           A,[BUFF7]
;           LD           B,A
;           CALL         TX
;
;           LD           A,[BUFF8]
;           LD           B,A
;           CALL         TX
;
;           LD           A,[BUFF5]
;           LD           B,A
;           CALL         TX
;           LD           A,[BUFF6]
;           LD           B,A
;           CALL         TX

```

```

;
;
;PERFORM BCC ON THE ASCII CODES WHICH ARE SENT
;
;
      XOR      A
      LD       IY,BUFF1
      XOR      [IY+0]
      XOR      [IY+1]
      XOR      [IY+2]
      XOR      [IY+3]
      XOR      [IY+4]
      XOR      [IY+5]
      XOR      [IY+6]
      XOR      [IY+7]
;
      LD       B,A           ;TX THE BCC
      CALL     TX
      LD       B,64          ;TX THE EOT "@"
      CALL     TX
;
; CHECK FOR ACK. FORM THE HOST COMPUTER
;
RX:      IN     A,[SIOC]      ;CHECK TO SEE IF ANY
      AND     2              ;CHARACTER RECEIVED?
      JP      Z,RX           ;IF NOT TRY AGAIN.
      IN     A,[SIOD]        ;IF YES READ IT.
      LD     B,6             ;IS IT ACK.?
      CP     B
      JP      NZ,KEEPON      ;IF NOT RETRANSMIT.
      RET              ;RET. IF IT WAS ACK.
;
;
; THIS IS "TX" SUBROUTINE.IT TRANSMITS THE ASCII
; CHARACTER IN THE "B" REG. CONTENTS OF A AND C
; WILL BE DESTROYED.
;
;
;
;
;
;
;
TX:      IN     A,[SIOC]      ;READ THE SI OC
      AND     1              ;MASK THE LSB
      JP      Z,TX           ;8251 NOT READY?
      LD     C,SIOD
      OUT     [C],B
      RET

```


THE PROGRAM WHICH CREATES TABLE1

[illegible]

THE PROGRAM WHICH CREATES TABLE2

"Z80" LIST XREF

;THIS IS NOT A PART OF THE SYSTEM PROGRAM. THIS PROGRAM IS ONLY
 ;USED TO CREATE "TABLE2". TABLE2 IS LINKED WITH THE SYSTEM
 ;SOFTWARE AND PROGRAMMED IN TO AN EPROM. WINDOW SIZE IS:
 ; 232 BY 400. SKIP FIRST 40 LINES AND 73 PIXELS

	ORG	0000H	
TABLE2	EQU	1000H	
LOW1	EQU	2FH	;FOR ODD FEILD START INFO.
HIGH1	EQU	89H	
STBIT1	EQU	01H	
LABIT1	EQU	07H	
LOW2	EQU	04CH	
HIGH2	EQU	89H	
NBYTE	EQU	30	
LINCTR	EQU	1FFFH	
FRAME	EQU	1FFEh	
LOW3	EQU	9EH	;END OF LINE FOR ODD FEILD
HIGH3	EQU	0C3H	
STBIT2	EQU	08H	
LABIT2	EQU	08H	
LOW4	EQU	0BAH	;END OF LINE FOR EVEN FEILD
HIGH4	EQU	0C3H	
NBYTE2	EQU	29	
NUMLIN	EQU	201	
BYTE1	EQU	29	;# OF BYTES TO PROCESS
BYTE2	EQU	30	
ADD1	EQU	38H	;#OF BYTES PER LINE
ADD2	EQU	39H	
SAVE	EQU	8000H	
STOD	EQU	001H	;FOR ODD
ENDOD	EQU	0FEH	;FOR ODD
STEV	EQU	0FFH	;FOR EVEN
ENDEV	EQU	0FFH	;FOR EVEN
;			
	LD	SP,0FFFFH	
	XOR	A	
	LD	[FRAME],A	
	LD	[LINCTR],A	;RESET LINE COUNTER
	LD	IY,TABLE2	;TABLE 2 START ADD. OF TABLE
	LD	L,LOW1	;L HAS LOW BYTE OF START ADD.
	LD	H,HIGH1	
	LD	B,STBIT1	;B HAS START NUMBER OF BITS
	EXX		
	LD	H,HIGH2	
	LD	L,LOW2	
	LD	D,STOD	
	LD	E,ENDOD	
	EXX		
	LD	E,LABIT1	;E HAS LAST NUMBER OF BITS
	LD	D,NBYTE	;D HAS # OF BYTES TO PROCESS
;			
STORE1:	LD	A,[LINCTR]	

```

                INC      A           ;INC. LINE COUNTER
                PUSH     BC
                LD        C,NUMLIN   ;CHECK SEE IF LINECOUNTER=150
                CP        C
                JP        NZ,CONT
;CHECK FOR EVEN FRAME
;
                LD        HL,FRAME

                XOR       A
                CP        [HL]
                JP        Z,EVEN
                POP       BC
                HALT
EVEN:           POP       BC
                INC      A
                LD        [FRAME],A
                LD        IY,TABLE2
                LD        BC,7
                ADD       IY,BC
                LD        L,LOW3
                LD        H,HIGH3
                LD        B,STBIT2
                EXX
                LD        H,HIGH4
                LD        L,LOW4
                LD        D,STEV
                LD        E,ENDEV
                EXX
                LD        E,LABIT2
                LD        D,NBYTE2
                LD        A,1
                LD        [LINCTR],A
                JP        CONT2
;
;
CONT:           LD        [LINCTR],A
                POP       BC
;
CONT2:         LD        [IY+0],L
                LD        [IY+1],H
                EXX
                LD        [IY+2],D
                LD        [IY+3],L
                LD        [IY+5],E
                EXX
                LD        [IY+6],D
                PUSH     BC
                LD        BC,14
                ADD       IY,BC
                POP      BC
;

```

```

;
LD      A,8
CP      B           ;CHECK FOR 8
JP      NZ,PT1
LD      B,1         ;SO IT WAS AN 8

LD      E,7
PUSH    BC
LD      BC,ADD1
ADD     HL,BC
POP     BC
LD      D,BYTE2
JP      STORE

;
;
PT1:    INC      B           ;WAS NOT 8
CP      B           ;CHECK FOR 8 AGAIN
JP      NZ,PT3
LD      D,BYTE1       ;IT WAS 8
LD      E,8
PUSH    BC
LD      BC,ADD2
ADD     HL,BC         ;PERFORM HL+38H----->HL
POP     BC
JP      STORE

;
;
;
PT3:    SCF           ;PERFORM 8-B----->E
LD      A,9
SBC     A,B
LD      E,A
LD      D,BYTE2
PUSH    BC
LD      BC,ADD2
ADD     HL,BC
POP     BC
JP      STORE

;
;FIND ADDRESS FOR END OF THE LINE
;
STORE:  LD      [SAVE],HL   ;SAVE HL
LD      A,D             ;SAVE D IN A
EXX
DEC     A
LD      C,A
XOR     A
LD      B,A
LD      HL,[SAVE]
ADD     HL,BC
EXX
LD      A,B
EXX

```

```
;
;FIND START MASK
;
          LD      B,A
          XOR     A
A1:       SCF
          RLA
          DEC     B
          JP      NZ,A1
          LD      D,A
          EXX
;
;FIND END MASK
;
          LD      A,E
          EXX
          LD      B,A
          XOR     A
A2:       SCF
          RR      A
          DEC     B
          JP      NZ,A2
          LD      E,A
          EXX
          JP      STORE1
;
;
          END
```

THE PUSH-TO-TEST ROUTINE

```

"Z80" LIST XREF      ORG          550H
                     EXT          INIT,SIOD,STACK
;
;THIS IS NMI ROUTINE WHICH IS USED TO DO A SELF TEST ROUTINE
;
                LD          SP,STACK
                CALL        INIT           ;INITILIZE 8251A
                LD          C,SIOD         ;8251 DATA BASE
                                           ;ADDRESS.
                LD          B,65           ;TX. "A" THRU "Z"
                LD          D,22           ;D CONTAINS THE # OF
START:          OUT        [C],B          ;TX THE CONTENT OF "B"
                DEC         D              ;LAST CHARACTER YET?
                JP         Z,NEXT
                LD          A,0FFH         ;DELAY LOOP SO 8251A
REPEAT:         DEC         A              ;HAS ENOUGH TIME TO
                JP         NZ,REPEAT       ;TX.
                JP         START
;
;CHECK OUT RAM
;
NEXT:           LD          A,0AAH         ;TRY THE FIRST TEST
                                           ;PATTERN = "AA"
                LD          B,0FFH
                CALL        CHECK          ;THIS SUB. WRITES THE
                                           ;PATTERN AND READ IT
                                           ;BACK FROM THE RAM
                                           ;FOR VERIFECATION.IN
                                           ;CASE OF A MISSMATCH
                                           ;IT WILL STAY IN AN
                                           ;INFINITE DELAY LOOP.
                LD          A,55H          ;GET THE NEXT TEST
                                           ;PATTERN.
                CALL        CHECK
                HALT                    ;RAMS HAVE BEEN
                                           ;CHECKED OUT SO GO
                                           ;TO "HALT" STATE SO
                                           ;"LED1" LIGHTS UP
                                           ;NOW.
;
;
;THIS SUBROUTINE WRITES A GIVEN TEST PATTERN
;WHICH IS INSIDE REGISTER "A" AND WRITES IT
;TO THE RAM LOCATIONS 8000H THROUGH FEFEH .
;AS SOON AS THE PATTERN IS WRITTEN IN TO A
;RAM LOCATION, IT WILL BE READ BACK TO TEST
;THAT RAM LOCATION. IN CASE OF A MISSMATCH
;THE PROGRAM WILL STAY IN AN INFINTIE DELAY
;LOOP.

```

```

;
CHECK:      LD      HL,8000H      ;HL IS MEMORY POINTER
            EX      AF,AF'
            LD      A,0FEH
CONT1:      EX      AF,AF'
            LD      [HL],A        ;WRITE THE PATTERN
            CP      [HL]          ;TEST THE RAM LOCATION
            JP      NZ,ERROR      ;STAY IN THIS LOOP IF
                                   ;IT FAILS.
            INC     HL            ;INCREMENT THE MEMORY
                                   ;POINTER
            EX      AF,AF'        ;CHECK FOR THE END
            CP      H             ;OF TESTING.
            JP      NZ,CONT1      ;IF NOT FINISHED GO
            CP      L             ;TO CONT1 OTHERWISE
            JP      NZ,CONT1      ;CONTINUE.
            EX      AF,AF'
            RET

```

APPENDIX B

THE FORMAT OF THE DIGITIZED PICTURE IN THE RAM

Memory :Bytes :Blocked

Address	Data			:Hex				
8000-07	CB	FF	FF	FF	FF	FF	FF	FF
8008-0F	FF	FF	FF	FF	FF	FF	FF	FF
8010-17	FF	FF	FF	FF	FF	FF	FF	FF
8018-1F	FF	FF	FF	FF	FF	FF	FF	FF
8020-27	FF	FF	FF	FF	FF	FF	FF	FF
8028-2F	FF	FF	FF	FF	FF	FF	FF	FF
8030-37	FF	FF	FF	FF	FF	FF	FF	FF
8038-3F	FF	FF	FF	FF	FF	FF	FF	FF
8040-47	FF	FF	FF	FF	FF	FF	FF	FF
8048-4F	FF	FF	FF	FF	FF	FF	FF	FF
8050-57	FF	FF	FF	FF	FF	FF	FF	FF
8058-5F	FF	FF	FF	FF	FF	FF	FF	FF
8060-67	FF	FF	FF	FF	FF	FF	FF	FF
8068-6F	FF	FF	FF	FF	FF	FF	FF	FF
8070-77	FF	FF	FF	FF	FF	FF	FF	FF
8078-7F	FF	FF	FF	FF	FF	FF	FF	FF
8080-87	FF	FF	FF	FF	FF	FF	FF	FF
8088-8F	FF	FF	FF	FF	FF	FF	FF	FF
8090-97	FF	FF	FF	FF	FF	FF	FF	FF
8098-9F	FF	FF	FF	FF	FF	FF	FF	FF
80A0-A7	FF	FF	FF	FF	FF	FF	FF	FF
80A8-AF	FF	FF	FF	FF	FF	FF	FF	FF
80B0-B7	FF	FF	FF	FF	FF	FF	FF	FF
80B8-BF	FF	FF	FF	FF	FF	FF	FF	FF
80C0-C7	FF	FF	FF	FF	FF	FF	FF	FF
80C8-CF	FF	FF	FF	FF	FF	FF	FF	FF
80D0-D7	FF	FF	FF	FF	FF	FF	FF	FF
80D8-DF	FF	FF	FF	FF	FF	FF	FF	FF
80E0-E7	FF	FF	FF	FF	FF	FF	FF	FF
80E8-EF	FF	FF	FF	FF	FF	FF	FF	FF
80F0-F7	FF	FF	FF	FF	FF	FF	FF	FF
80F8-FF	FF	FF	FF	FF	FF	FF	FF	FF
8100-07	FF	FF	FF	FF	FF	FF	FF	FF
8108-0F	FF	FF	FF	FF	FF	FF	FF	FF
8110-17	FF	FF	FF	FF	FF	FF	FF	FF
8118-1F	FF	FF	FF	FF	FF	FF	FF	FF
8120-27	FF	FF	FF	FF	FF	FF	FF	FF
8128-2F	FF	FF	FF	FF	FF	FF	FF	FF
8130-37	FF	FF	FF	FF	FF	FF	FF	FF
8138-3F	FF	FF	FF	FF	FF	FF	FF	FF
8140-47	FF	FF	FF	FF	FF	FF	FF	FF
8148-4F	FF	FF	FF	FF	FF	FF	FF	FF
8150-57	FF	FF	FF	FF	FF	FF	FF	FF
8158-5F	FF	FF	FF	FF	FF	FF	FF	FF
8160-67	FF	FF	FF	FF	FF	FF	FF	FF
8168-6F	FF	FF	FF	FF	FF	FF	FF	FF
8170-77	FF	FF	FF	FF	FF	FF	FF	FF
8178-7F	FF	FF	FF	FF	FF	FF	FF	FF

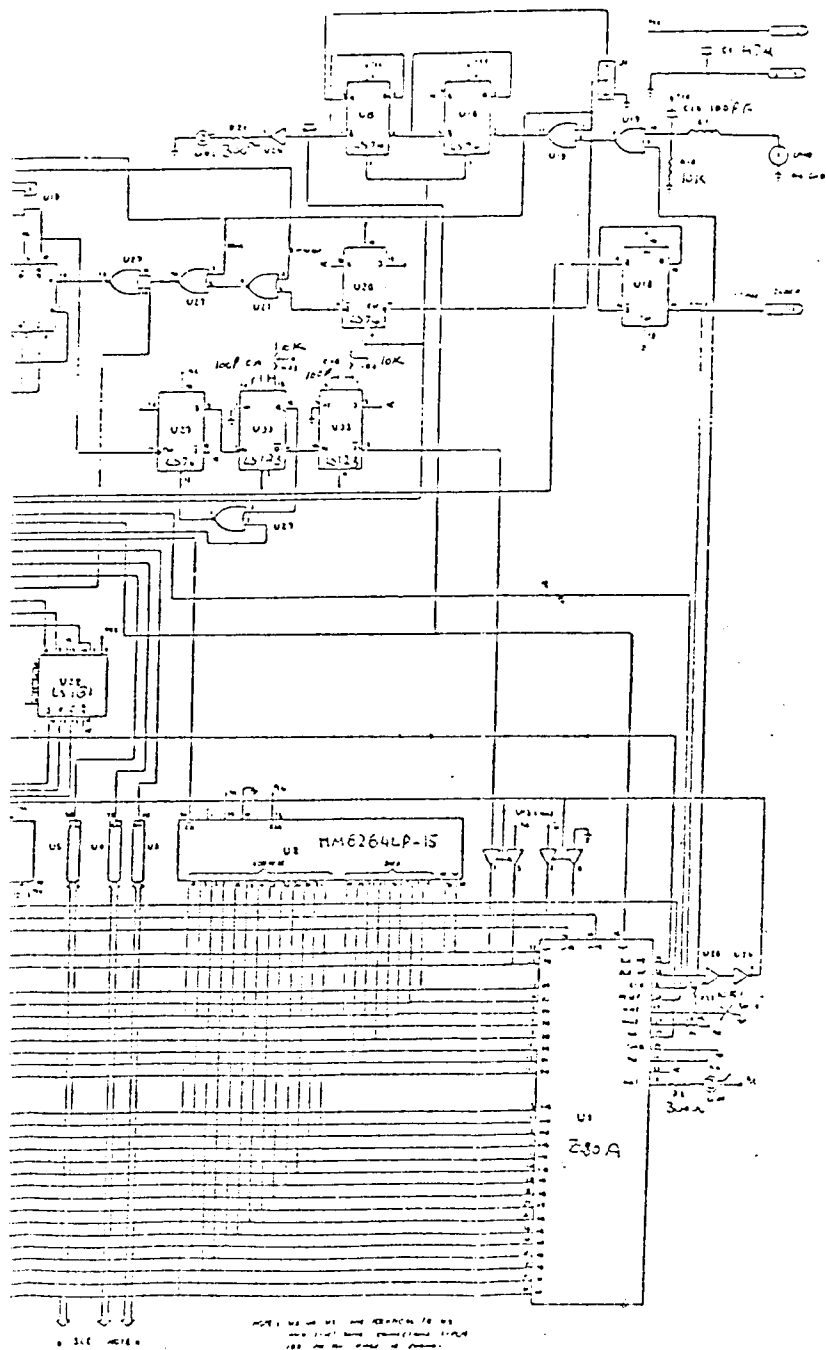
8660-67	00	00	00	00	00	00	00	00
8668-6F	00	00	00	00	00	00	00	00
8670-77	00	00	FF	FF	FF	FF	FF	FF
8678-7F	FF	FF	FF	F8	00	00	00	00
8680-87	00	00	00	00	00	00	00	00
8688-8F	00	00	00	00	00	00	00	00
8690-97	00	00	00	00	00	00	00	00
8698-9F	00	00	00	00	00	00	00	00
86A0-A7	00	00	00	00	00	00	00	00
86A8-AF	00	00	01	FF	FF	FF	FF	FF
86B0-B7	FF	FF	FF	FF	F0	00	00	00
86B8-BF	00	00	00	00	00	00	00	00
86C0-C7	00	00	00	00	00	00	00	00
86C8-CF	00	00	00	00	00	00	00	00
86D0-D7	00	00	00	00	00	00	00	00
86D8-DF	00	00	00	00	00	00	00	00
86E0-E7	00	00	00	03	FF	FF	FF	FF
86E8-EF	FF	FF	FF	FF	FF	E0	00	00
86F0-F7	00	00	00	00	00	00	00	00
86F8-FF	00	00	00	00	00	00	00	00
8700-07	00	00	00	00	00	00	00	00
8708-0F	00	00	00	00	00	00	00	00
8710-17	00	00	00	00	00	00	00	00
8718-1F	00	00	00	00	07	FF	FF	FF
B9FE-05	FF	FF	FF	FF	FF	FF	FF	FF
BA06-0D	F8	00	00	00	00	00	00	00
BA0E-15	00	00	00	00	00	00	00	00
BA16-1D	00	00	00	00	00	00	00	00
BA1E-25	00	00	00	00	00	00	00	00
BA26-2D	00	00	00	00	00	00	00	00
BA2E-35	00	00	00	00	00	00	00	01
BA36-3D	FF	FF	FF	FF	FF	FF	FF	FF
BA3E-45	FF	F0	00	00	00	00	00	00
BA46-4D	00	00	00	00	00	00	00	00
BA4E-55	00	00	00	00	3F	FF	FF	FF
BA56-5D	FF	FF	FF	FF	FF	FF	FF	FF
BA5E-65	FF	FF	FF	FF	FF	FF	FF	FF
BA66-6D	FF	FF	FF	FF	FF	FF	FF	FF
BA6E-75	FF	FF	FF	FF	FF	FF	FF	FF
BA76-7D	FF	FF	FF	FF	FF	FF	FF	FF
BA7E-85	FF	FF	FF	FF	FF	FF	FF	FF
BA86-8D	FF	FF	FF	FF	FF	FF	FF	FF
BA8E-95	FF	FF	FF	FF	FF	FF	FF	FF
BA96-9D	FF	FF	FF	FF	FF	FF	FF	FF
BA9E-A5	FF	FF	FF	FF	FF	FF	FF	FF
BAA6-AD	FF	FF	FF	FF	FF	FF	FF	FF
BAAE-B5	FF	FF	FF	FF	FF	FF	FF	FF
BAB6-BD	FF	FF	FF	FF	FF	FF	FF	FF
BABE-C5	FF	FF	FF	FF	FF	FF	FF	FF
BAC6-CD	FF	FF	FF	FF	FF	FF	FF	FF
BACE-D5	FF	FF	FF	FF	FF	FF	FF	FF

BC76-7D	FF	FF	FF	FF	FF	FF	FF	FF
BC7E-85	FF	FF	FF	FF	FF	FF	FF	FF
BC86-8D	FF	FF	FF	FF	FF	FF	FF	FF
BC8E-95	FF	FF	FF	FF	FF	FF	FF	FF
BC96-9D	FF	FF	FF	FF	FF	FF	FF	FF
BC9E-A5	FF	FF	FF	FF	FF	FF	FF	FF
BCA6-AD	FF	FF	FF	FF	FF	FF	FF	FF
BCAE-B5	FF	FF	FF	FF	FF	FF	FF	FF
BCB6-BD	FF	FF	FF	FF	FF	FF	FF	FF
BCBE-C5	FF	FF	FF	FF	FF	FF	FF	FF
BCC6-CD	FF	FF	FF	FF	FF	FF	FF	FF
BCCE-D5	FF	FF	FF	FF	FF	FF	FF	FF
BCD6-DD	FF	FF	FF	FF	FF	FF	FF	FF
BCDE-E5	FF	FF	FF	FF	FF	FF	FF	FF
BCE6-ED	FF	FF	FF	FF	FF	FF	FF	FF
BCEE-F5	FF	FF	FF	FF	FF	FF	FF	FF
BCF6-FD	FF	FF	FF	FF	FF	FF	FF	FF
BCFE-05	FF	FF	FF	FF	FF	FF	FF	FF
BD06-0D	FF	FF	FF	FF	FF	FF	FF	FF
BD0E-15	FF	FF	FF	FF	FF	FF	FF	FF
BD16-1D	FF	FF	FF	FF	FF	FF	FF	FF
BD1E-25	FF	FF	FF	FF	FF	FF	FF	FF
BD26-2D	FF	FF	FF	FF	FF	FF	FF	FF
BD2E-35	FF	FF	FF	FF	FF	FF	FF	FF
BD36-3D	FF	FF	FF	FF	FF	FF	FF	FF
BD3E-45	FF	FF	FF	FF	FF	FF	FF	FF
BD46-4D	FF	FF	FF	FF	FF	FF	FF	FF
BD4E-55	FF	FF	FF	FF	FF	FF	FF	FF
BD56-5D	FF	FF	FF	FF	FF	FF	FF	FF
BD5E-65	FF	FF	FF	FF	FF	FF	FF	FF
BD66-6D	FF	FF	FF	FF	FF	FF	FF	FF
BD6E-75	FF	FF	FF	FF	FF	FF	FF	FF
BD76-7D	FF	FF	FF	FF	FF	FF	FF	FF
BD7E-85	FF	FF	FF	FF	FF	FF	FF	FF
BD86-8D	FF	FF	FF	FF	FF	FF	FF	FF
BD8E-95	FF	FF	FF	FF	FF	FF	FF	FF
BD96-9D	FF	FF	FF	FF	FF	FF	FF	FF
BD9E-A5	FF	FF	FF	FF	FF	FF	FF	FF
BDA6-AD	FF	FF	FF	FF	FF	FF	FF	FF
BDAE-B5	FF	FF	FF	FF	FF	FF	FF	FF
BDB6-BD	FF	FF	FF	FF	FF	FF	FF	FF
BDBE-C5	FF	FF	FF	FF	FF	FF	FF	FF
BDC6-CD	FF	FF	FF	FF	FF	FF	FF	FF
BDCE-D5	FF	FF	FF	FF	FF	FF	FF	FF
BDD6-DD	FF	FF	FF	FF	FF	FF	FF	FF
BDDE-E5	FF	FF	FF	FF	FF	FF	FF	FF
BDE6-ED	FF	FF	FF	FF	FF	FF	FF	FF
BDEE-F5	FF	FF	FF	FF	FF	FF	FF	FF
BDF6-FD	FF	FF	FF	FF	FF	FF	FF	FF
BDFE-05	FF	FF	FF	FF	FF	FF	FF	FF
BE06-0D	FF	FF	FF	FF	FF	FF	FF	FF
BE0E-15	FF	FF	FF	FF	FF	FF	FF	FF

BFB6-BD	00	00	00	00	00	00	00	00
BFBE-C5	00	00	00	00	00	03	FF	FF
BFC6-CD	FF	FF	FF	FF	FF	FF	FF	E0
BFCE-D5	00	00	00	00	00	00	00	00
BFD6-DD	00	00	00	00	00	00	00	00
BFDE-E5	00	00	00	00	00	00	00	00
BFE6-ED	00	00	00	00	00	00	00	00
BFEE-F5	00	00	00	00	00	00	00	00
BFF6-FD	00	00	00	00	00	00	07	FF
BFFE-05	FF	FF	FF	FF	FF	FF	FF	FF
C006-0D	C0	00	00	00	00	00	00	00
C00E-15	00	00	00	00	00	00	00	00
C016-1D	00	00	00	00	00	00	00	00
C01E-25	00	00	00	00	00	00	00	00
C026-2D	00	00	00	00	00	00	00	00
C02E-35	00	00	00	00	00	00	00	0F
C036-3D	FF	FF	FF	FF	FF	FF	FF	FF
C03E-45	FF	80	00	00	00	00	00	00
C046-4D	00	00	00	00	00	00	00	00
C04E-55	00	00	00	00	00	00	00	00
C056-5D	00	00	00	00	00	00	00	00
C05E-65	00	00	00	00	00	00	00	00
C066-6D	00	00	00	00	00	00	00	00
C06E-75	1F	FF	FF	FF	FF	FF	FF	FF
C076-7D	FF	FF	00	00	00	00	00	00
C07E-85	00	00	00	00	00	00	00	00
C086-8D	00	00	00	00	00	00	00	00
C08E-95	00	00	00	00	00	00	00	00
C096-9D	00	00	00	00	00	00	00	00
C09E-A5	00	00	00	00	00	00	00	00
C0A6-AD	00	3F	FF	FF	FF	FF	FF	FF
C0AE-B5	FF	FF	FE	00	00	00	00	00
C0B6-BD	00	00	00	00	00	00	00	00
C0BE-C5	00	00	00	00	00	00	00	00
C0C6-CD	00	00	00	00	00	00	00	00
C0CE-D5	00	00	00	00	00	00	00	00
C0D6-DD	00	00	00	00	00	00	00	00
C0DE-E5	00	00	7F	FF	FF	FF	FF	FF
C0E6-ED	FF	FF	FF	FC	00	00	00	00
C0EE-F5	00	00	00	00	00	00	00	00
C0F6-FD	00	00	00	00	00	00	00	00
C0FE-05	00	00	00	00	00	00	00	00
C106-0D	00	00	00	00	00	00	00	00
C10E-15	00	00	00	00	00	00	00	00
C116-1D	00	00	00	FF	FF	FF	FF	FF

APPENDIX C

THE COMPLETE SCHEMATIC OF THE TRASH-METER HARDWARE



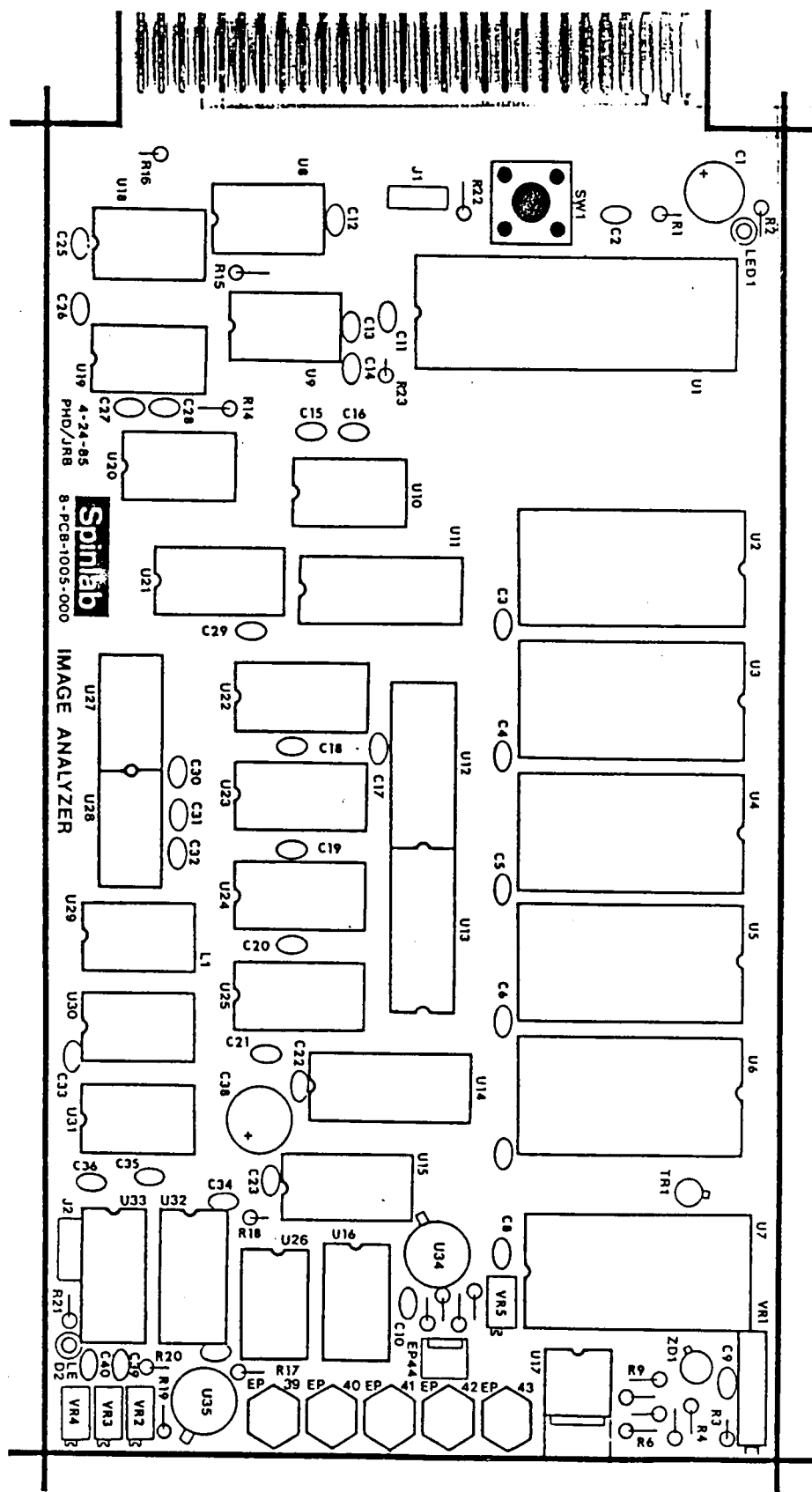


Figure 44. Trash-meter Circuit Board

VITA

Peyman Hosseini Dehkordi was born in Sary, Iran on November 18, 1960. He attended elementary schools in Tehran and was graduated from Dr. Hashtroudi High School in June 1978. The following September he entered The Elm va Sanat University in Tehran, Iran.

He came to The United States on May 1979 for the first time. He attended The University of Bridgeport in Connecticut for Three quarters. Then, he came to the University of Tennessee in Knoxville to continue his undergraduate program. He received the Bachelor of Science with a major in Electrical Engineering in December 1981. In the winter of 1982 he accepted two research projects with the University of Tennessee and began study toward a Master's Degree in Electrical Engineering. This degree will be awarded in June 1985.

The author is a member of Tau Beta Pi engineering honor society. He has also been employed by the SPIN LAB one year prior to his graduation.