1-1-1984

Design and construction of a Z80-based microcomputer system and associated monitor program.

Douglas Howard Rhyner

DESIGN AND CONSTRUCTION OF A Z80-BASED MICROCOMPUTER SYSTEM AND ASSOCIATED MONITOR PROGRAM

by

Douglas Howard Rhyner

A Thesis

Presented to the Graduate Committee

of Lehigh University

in Candidacy for the Degree of

Master of Science

in

Electrical Engineering

Lehigh University

1984

Certificate of Approval

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

5/9/84
Date

Carl S. Holzinger
Professor in Charge

Eric D. Thompson
Department Chairman

Table of Contents

List of Figures

Abstract

This thesis describes the design and construction of a Z80-based microcomputer system, and the design and implementation of the monitor software required to support this microcomputer. It contains the schematics, timing diagrams and hardware connection information which are required to produce a working model of this microcomputer system. The system contains a CPU board connected to an ASCII keyboard, a memory board containing 62 K of RAM memory and 2 K of ROM memory (for permanent storage of the monitor and some user programs), and a video board that produces ASCII characters and color graphics when connected to an ordinary color television. It also contains the Z80 code and detailed descriptions of the monitor software, which provides the user with the ability to enter and modify programs in the RAM memory, examine and execute these programs, and run self tests on the hardware.

I) INTRODUCTION

This thesis deals with the design and construction of a microcomputer system with a functional, user-friendly microcomputer operating system (monitor). The hardware design was done by determining what features the computer system should contain, selecting components that could provide these features, and designing the system and support circuits around these components. The system was assembled using wire-wrap sockets mounted on general purpose S-100 Bus compatible circuit boards. The monitor

was written exclusively for this hardware, in order to provide it with all the required functions, such as data entry or modification, program examination and execution, and self tests for the hardware. The following text has been divided into two major sections; the first dealing with the microcomputer system hardware, and the second with the operating system written to allow full use of this hardware.

II) HARDWARE

II-A) HARDWARE OVERVIEW

The hardware for this microcomputer system was designed with the following requirements in mind. The CPU should be a state of the art 8-bit microprocessor, capable of supporting 64K of memory and capable of input and output functions, including supporting an ASCII keyboard. The system should be able to operate with a standard television set, and generate both alphanumeric and color graphic characters. The memory should have a refresh that is transparent to the user and that cannot be accidentally halted (as in a Wait State). The system boards should be connected using the I.E.E.E. standard S-100 bus.

The system design was done by first selecting the major components that could be used to satisfy the above requirements, and then designing the peripheral circuits required to support them. The system hardware consists of three main sections: the CPU board, containing the Zilog Z80 microprocessor and RCA CDP1852 Byte-Wide (keyboard) input/output port; the Video board, containing the Motorola MC6847 Video Display Generator, Motorola MC1372 color TV

modulator, and the four one K by four bit static RAM memories; and the Memory board, containing the 64K dynamic RAM memories, the Intel 3242 Address Multiplexer and Refresh Counter, and the Intel 2716 EPROM. These three sections are discussed in the following paragraphs. Please refer to the appropriate schematic drawing (figures VIII, IX and X) for each section.

The boards are each supplied with seven volt regulated power through the S-100 bus. Each board converts this to five volts (all the Integrated Circuits used require five volts only) by passing it through a 7805 voltage regulator. A large filter capacitor (10 uF) is provided for the input and output of each regulator, and smaller filter capacitors (.1 uF) are scattered throughout each of the boards.

Status and control signals which are active when low (in the logic zero state) are indicated by a slash following the signal name. For example, MREQ/ is the signal thats indicates a memory access is in progress when it goes low.

II-B) CPU BOARD

This microcomputer system was built around a Zilog Z80 microprocessor (chip number 33 on the schematic). All outputs are buffered to protect the Z80. The reset, status and control signals are discussed in the following paragraphs. Further information on the Z80 microprocessor is available from the "Z80-CPU Technical Manual", available from Zilog.

Chip 29B, a 74LS123 Retriggerable Monostable Multivibrator, provides a 125 nanosecond Reset pulse to the Z80 when triggered. The 51K resistor and 1 Uf capacitor connected to the B input of the 74LS123 provide for the required reset on power-up. The 74LS123 will be triggered when the 1 Uf capacitor has been charged up to a level greater than two volts, approximately 0.02 seconds after power is applied. The Reset switch, also connected to the B input of the 74LS123, simulates a power-up condition when pressed. This provides a manual reset that can be used to restart the Z80 at location 0000 whenever required.

There are three LEDs that continuously monitor the status of the Z80. A red LED is connected to the Halt/ signal that lights whenever the Z80 has executed a halt instruction. This usually indicates that the Z80 is

waiting for input from the ASCII keyboard, that generates a maskable interrupt to location 0038 Hex. Another red LED is connected to the Z80s Wait/ line, that is lit when the Wait/ signal is low and the Z80 is paused. The third indicator is a green LED connected to the Z80s M1/ signal that lights when the Z80 is fetching an instruction, verifying that the CPU is still functioning.

Input and Output

Input and output is controlled by chip 30 on the CPU board, a 74LS138 3-to-8 Line Decoder/Demultiplexer. Chip 30 is enabled whenever IORQ/ and A7 are at a logical zero state and A6 is at a logic one state. This happens when the microprocessor executes an input or output instruction to a port with an address between 40 Hex and 7F Hex. A4, A5 and the Read/ signal select which of the eight outputs of chip 30 is selected. Four will be active during certain input instructions, and four during output instructions.

When the Z80 does an output to port 7X Hex (where X is any pattern of bits) chip 30 toggles Y7 to the low state, triggering chip 29A, a 74LS123 Retriggerable Monostable Multivibrator. Chip 29A uses a 300K resistor and a 1 UF capacitor for timing, which produces a 0.14 second pulse when triggered. This pulse activates the Pizeo-electric

buzzer, producing a short beep whenever an output to port 7X is executed.

The keyboard is a full ASCII keyboard, set up to use only capital letters. The keyboard was originally from a Digital Equipment Corporation "Decwriter I" terminal, and is based on an SMC KR2376-17 Keyboard Encoder ROM and Control chip. When any of the keys is pressed, the keyboard generates an interrupt to the Z80 and latches the data into chip 31, the RCA CDP1852 Byte-Wide I/O Port. When the Z80 responds to the interrupt it will execute an input instruction from port 7X. This input instruction will cause chip 30 to toggle Y6 to the low state, enabling the chip enable of chip 31. Chip 31 then places data from the keyboard onto the data bus, from where it is read into the Z80s accumulator.

Clock

The Z80 is operated at a clock rate of 1.8 Megahertz. Two of the inverters from chip 16 are connected to a 1.8 Megahertz crystal to generate the clock pulse. The two 330 ohm resistors connected between the input and output of the inverters bias the inverters into their region of linear operation. The 680 pf capacitor provides a small delay between the output of the first inverter and the input of

the next.

Chip 24 is a 74LS365 buffer which is used to "clean up" the clock signal, making it more of a square wave. The 5.6K pull-up resistor is used to force the logic one state of the clock signal to five volts. Without it the clocks logic one state would be the same level as the TTL gate, approximately 3.4 volts, which is insufficient for correct operation of the Z80.

Address Bus, Address Display and Data Bus

Chips 22, 23 and 24 are 74LS365 Buffers, that connect the Z80 to the S-100 bus. Four Hewlett Packard 5082-7395 Hexadecimal LED Displays with Latches (DP0 through DP3) are used to monitor the information being passed to the bus, constantly displaying the current address. The address is latched into the displays whenever the Z80s M1/ status signal goes low (indicating that the Z80 is fetching an instruction or responding to an interrupt) guaranteeing that the information being displayed is the current address of the Z80s program counter.

Chips 25 and 26, 74LS365 Buffers, and chips 27 and 28, 74LS367 Buffers, connect the Z80 to the data bus. Chips 26 and 27 are active only when the Z80s Read/ signal is low, and chips 25 and 28 are active only when the Write/ signal

is low. This arrangement prevents data from being placed onto the bus or to the Z80 when it shouldn't be, and allows greater control over the data bus.

II-C) VIDEO BOARD

Processor

The video board is designed around chip 32, a Motorola MC6847 Video Display Generator. The Video Display Generator (VDG) can display 64 different ASCII characters, and can produce color graphics in eight different colors. The ASCII characters can be displayed in red or green, depending on the setting of the CSS DIP switch (the top switch, located to the right of chip 32 on the drawing, connected to pin 39 of the VDG).

The VDG reads data from memory and produces a composite video signal. The VDG will continuously scan the video memory, process the information, and pass the required video signals to the modulator.

The VDG can generate 64 different alphanumeric characters from information stored in an internal character ROM. The television screen is divided into 512 displayable character locations, arranged as 16 lines, each containing 32 characters. When displayed, each character location occupies an eight by twelve dot matrix box. The actual character consists of a five by seven dot matrix, with the remaining dots being used for character spacing.

Because only six bits are required to uniquely identify each of the 64 characters, the two high order bits can be used inform the VDG whether the character should be displayed normally or in a modified form. Bit six of each character determines if the character will be displayed in the Inverse Video mode, a mode that can be used to highlight certain information. Bit six is therefore tied to both the D6 and INV inputs to the VDG, as shown in the Video Board drawing.

The VDG can be used in several different modes, only one of which (the Alpha/Graphic mode) generates both graphics and alphanumeric characters. The Alpha/Graphic mode is the standard mode, which is normally used by the system. When the VDG is using this mode data bit seven determines whether the information stored in memory will be displayed as a graphics character, or as an alphanumeric. If bit seven is a "zero", the character is an alphanumeric; otherwise the information will be displayed as a graphics character. The graphics characters divide each of the 16 by 32 screen locations into four displayable sections. Bits 0 through 3 determine which of the sections are "on" (activated), and bits 4 through 6 determine which of the eight colors the "on" sections are displayed as. Sections that are not activated are displayed in black.

The Alpha/Graphics mode requires only 512 of the 2048 video memory locations, the remainder of the video memory is used when more detailed graphics is required. Graphics designs can be produced in more detail by using one of the "pure graphics" modes, where the screen is divided up into a number of small squares, commonly called pixels. The screen can be divided into individually addressable pixels which are 64 rows by 64 columns, 64 rows by 128 columns, or 96 rows by 128 columns (the 96 by 128 mode uses the full two K of memory). The pure graphics modes are selected using the DIP switches connected to the VDG. The switch connected to pin 35 of the VDG is closed (grounding pin 35) for the normal Alpha/Graphics mode, and is open (placing 5 volts on pin 35) for the pure graphics modes. The remaining two DIP switches are active only when pure graphics is selected. These switches select which of the three graphics modes described above is selected. A zero-zero selects the 64 by 64 mode, one-zero selects the 64 by 128 mode, and one-one selects the 96 by 128 mode.

Modulator

Chip 17 is an MC1372 Color TV Video Modulator, that is used to generate an RF TV signal when provided with baseband color-difference and luminance video signals. The MC1372 is made for operation with the MC6847 Video Display

Generator, which provides these video signals. The MC1372 modulates the video signal to produce an ordinary television signal, and can broadcast anywhere from channel two to channel six. If a small antenna is connected to the video output of the MC1372, no direct connection between the computer and television is required. The MC1372 can be tuned to a frequency where there is no local television station, and a satisfactory video signal can be received at a distance of 20 to 40 feet.

The MC1372 contains a Chrominance Oscillator and Clock Driver, a Lead and Lag Network, a Chroma Modulator, an RF Oscillator, and an RF modulator. Further information on the specifications and internal operation is available from the Motorola MC1372 data sheets.

Pins 1, 5, 6, 7 and 9 connect directly to the video processor, providing the processor with its required clock signal, and the modulator with the required color and luminance signals. Pin 2 is for the MC1372s oscillator. A 3.579545 MHZ crystal connected to this pin provides the proper clock frequency for the MC1372. Pin 3 is for setting the duty cycle of the clock output. Pin 8 feeds the Chroma Modulator output through the appropriate gain reduction resistor and coupling capacitor, into the Chrominance Input.

Pins 13 and 14 are for an RF Tank circuit, which is used to select the television channel of the output signal. Tuning the .1UH inductor will vary the output between channel two and channel six. The 240 ohm resistors control the amplitude of the output signal. Pin 12 is the modulated RF signal in the form required by normal television sets. The impedance-matching resistor from this pin to +5 volts should match the characteristic impedance of the output cable being used (300 ohms in this case) to prevent feedback.

Memory, Memory Select

The Video board contains two K of eight-bit words of Static memory, memory addresses F800 Hex to FFFF Hex, which is used to hold the information to be displayed on the television screen. Each memory chip contains one K of four bit words, so two pairs of chips are required (i.e., chips 9 and 10 are the first one K of memory, chip 9 containing D0 - D3, and chip 10 containing D4 - D7). These memories are written to only by the Z80, and read only by the MC6847 Video Display Generator, as mentioned earlier.

Chip 13 is the video memory select control. When Write/ is low and Addresses A11 through A15 are high, the Z80 is outputting the data to be displayed on the television

screen, and therefore is writing to the video memory. The video memory select signal produced by chip 13 activates the 74LS365 Tri-state buffers (chips 19,20 and 21) which gate the data and address information from the Z80 to the video memory chips. The video memory select signal also lowers the WE/ signal on the memory chips, allowing them to receive information from the Z80, and activates the MS/ signal of the VDG (pin 12), which causes its address and data outputs to go tri-state (relinquishing control over the memory chips). The last function of the video memory select signal is to switch control of the video memory chip select line to the address bus as explained in the following section.

Chip select

Chip 15, a 74LS08 quad two-input And gate, is the actual video memory chip select. When any one of the inputs drops to the low state, the output of that gate will select the corresponding two memory chips. The memory select signal determines which pair of chip 14s Nand gates (the upper or lower pair as shown in the schematic) has control of chip 15 (the video memory chip select).

If the memory select signal is high, the outputs of the bottom two Nand gates are forced high (disabled), while the

top two are free to use VA10 (from the Video Display Generator) to control the chip select And gates. If the memory select signal is low, the top two Nand gates are disabled, and the bottom two pass A10 (from the address bus) through to control the chip select And gates.

II-D) MEMORY BOARD

ROM

Chip 33 is a 2716 two K by eight ultravioletly Erasable Reprogrammable Read Only Memory (EPROM) that contains the monitor and utility programs for the system. It is memory mapped in at addresses 0000 to 07FF Hex, and is only active when the Z80 is fetching instructions from these addresses. The ROM is never selected when the RAM or Video RAM memory is selected. When the system is reset or powered-up, the Z80 automatically begins executing the code contained in this EPROM at location 0000 Hex.

The monitor and utility programs fill only one K of the EPROM, leaving the second K of memory free for permanent storage of user programs. The contents of the EPROM can be easily modifyed with the proper equipment.

RAM

Chips 24 through 31 are industry standard 64K Dynamic RAM memory chips. Eight of these memory chips, each of which contain 65,535 (64K) one-bit words, are used in parallel. The memory chips are shown in the memory drawing as one block, since all connections except the data line are common to each memory chip. There are only eight address

inputs to the memory chips; the current read or write address is multiplexed into the memory chips by chip 32, the Intel 3242 Address Multiplexer and Refresh Counter.

The memory chips are Dynamic RAMs, and therefore must periodically be refreshed in order for them to retain their data. They use a RE/ only refresh, in which lowering the RE/ signal will refresh the entire row currently addressed by the address bus. The memories are refreshed as if they were 16K memories (only the first 128 refresh cycles are required), which allows the use of the Intel 3242 Address Multiplexer and Refresh Counter. The maximum refresh period for the memories is 4.4 ms, therefore refreshing one row after each instruction fetch (the standard refresh scheme used by the Z80) is more than adequate. At the clock rate used by this system, 1.8 Megahertz, the refresh period is 0.53 ms on the average, and 0.71 ms worst case. The actual rate depends on the instructions being executed by the Z80.

In order to read from or write to the dynamic RAM chips, the control signals must be applied in the correct sequence. The row address is first presented to the RAM address lines, RE/ is lowered, the column address is presented to the RAM, and CE/ is lowered. This sequence is discussed in more detail in the Memory Timing sections.

Timing diagrams for the read, write and refresh functions are presented in the following pages.

Memory Read Timing

The memory read (or instruction fetch) function requires three clock cycles (T states) to complete. The control signals generated by the Z80 and the data-in requirement are shown below. Each of the 1.8 Megahertz clock cycles is 0.556 microseconds long.

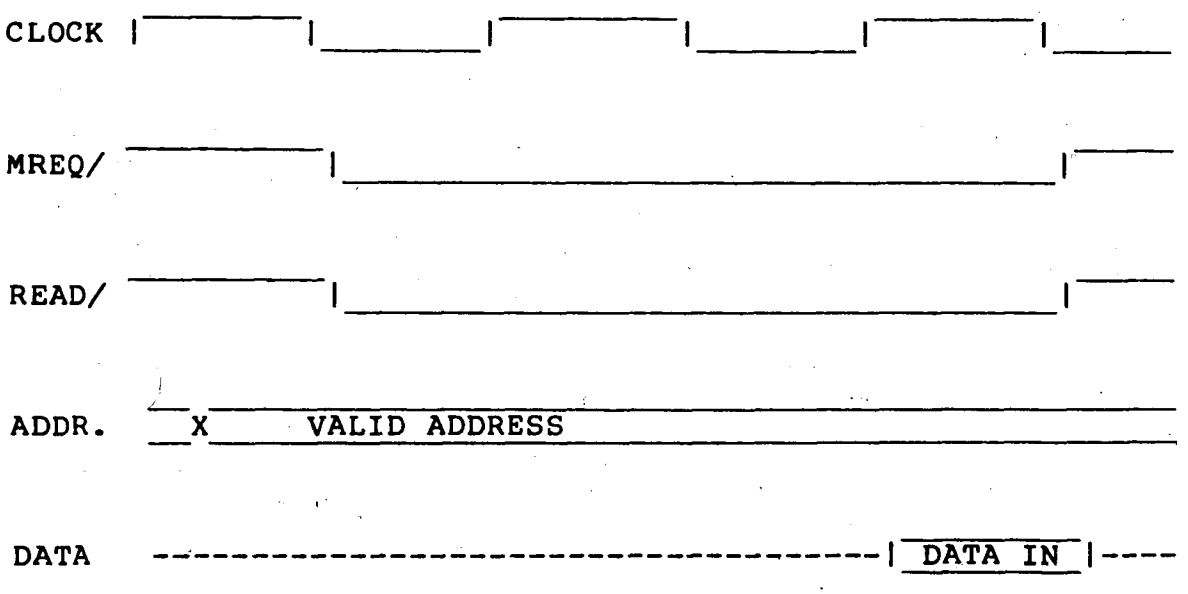

Figure I - Memory Read Timing Diagram

The major memory transactions all occur within the first clock cycle. This clock cycle and the memory control signals are explained here and shown in Figure III.

The read (or write) cycle is initiated by a drop in the MREQ signal. This signal passes through a small delay circuit (created by connecting two inverters from chip 17), and becomes the Row Enable (RE/) input to the RAM chips. The delay is to guarantee that the write pulse (in a write cycle only) is presented to the RAM chips early enough, and the amount of delay is not critical. The high-to-low transition of the RE/ line latches the eight bit row address from chip 32 (the Intel address multiplexer, or MUX) into the RAM chips, and signals the MUX that a memory access cycle has begun. There are three critical timing requirements for a memory access cycle that must not be violated. These delays are based on the requirements of both the Intel MUX and the dynamic RAM memories, and are listed in the following figure.

Start and End Event	Minimum Delay	Maximum Delay
RE/ low till CE/ low	30 ns	70 ns
RE/ low till Row/Column/ low	12 ns	50 ns
Row/Column/ low till CE/ low	20 ns	58 ns

Figure II - Critical MUX Timing Requirements

The row address is the information normally present on the MUX address lines, so the next step after RE/ goes low is to lower the Row/Column/ input of the MUX, to indicate that the column address should now be made available. This is accomplished by passing the RE/ signal through two gates (18 and 17) to the Row/Column/ input, causing the column address to be placed on the mux address lines. The RE/ low to Row/Column/ low delay is approximately 17 ns. By examining the data sheets for gates 17 and 18, we find that the maximum delay is 20.5 ns and the minimum delay is 14 ns, both of which are within the limits shown in figure II. The logic zero signal that has just appeared at the Row/Column/ input of the MUX is then passed through another delay circuit, which consists of two of chip 19s And gates, to the Chip Enable (CE/) input of the RAM chips. When this CE/ line goes low the eight bit column address is latched

from the MUX into the RAM memories. The maximum and minimum delays for these two And gates (from the 74LS08 data sheets) are 40 ns and 20 ns, respectively. These worst case delays are both within the limits outlined in figure II. The maximum and minimum delays between RE/ low and CE/ low are found from the above data to be 34 ns and 60.5 ns, both of which are within the limits shown in figure II. The data out of the RAM chips is available on the Q line of the RAMS 100 ns after the CE/ signal goes low.

These transitions are shown graphically in the following diagram (Figure III).

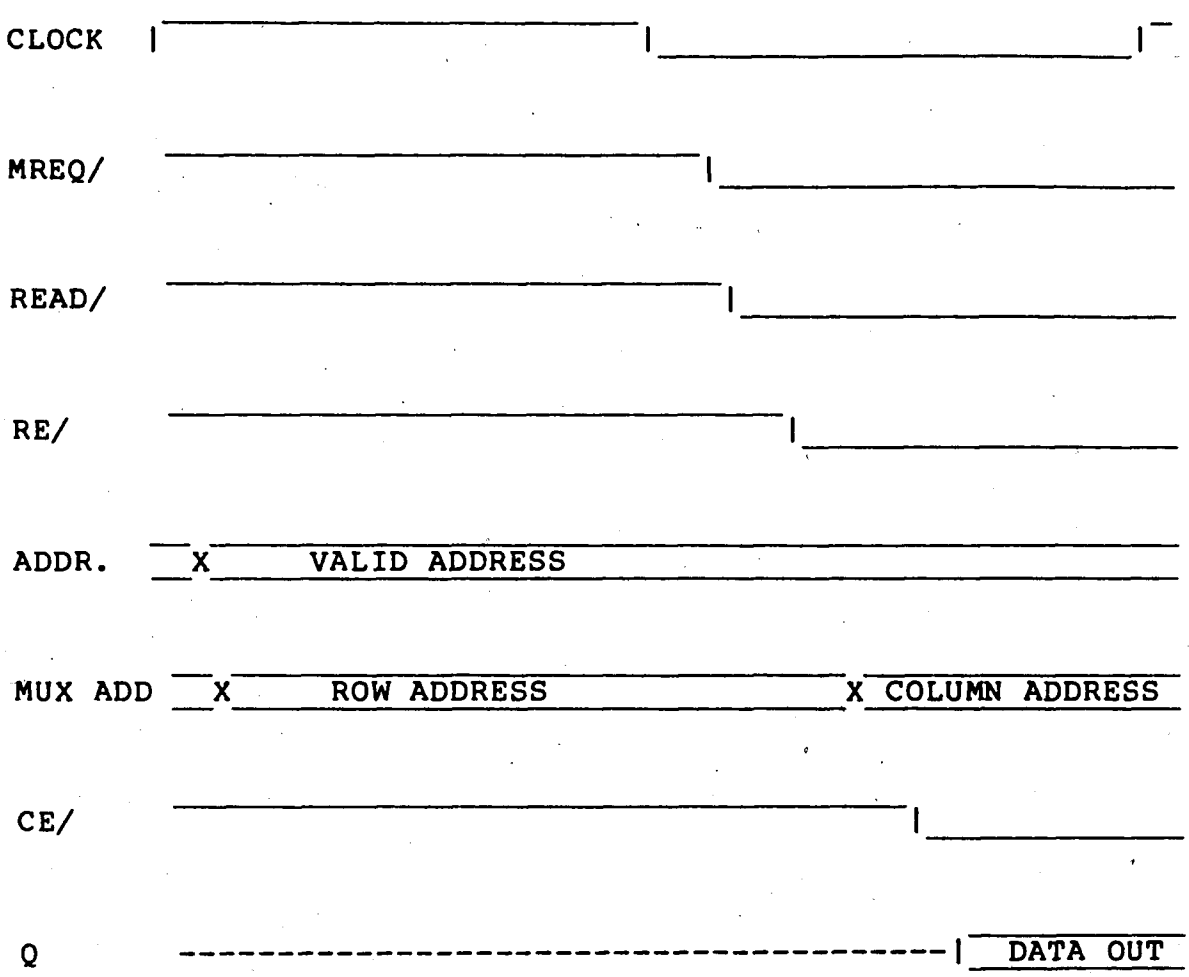

Figure III - Expanded Memory Read Timing Diagram

Memory Write Timing

The memory write function requires three clock cycles to complete. The control signals generated by the Z80 are shown below. Each of the clock cycles is 0.556 microseconds.

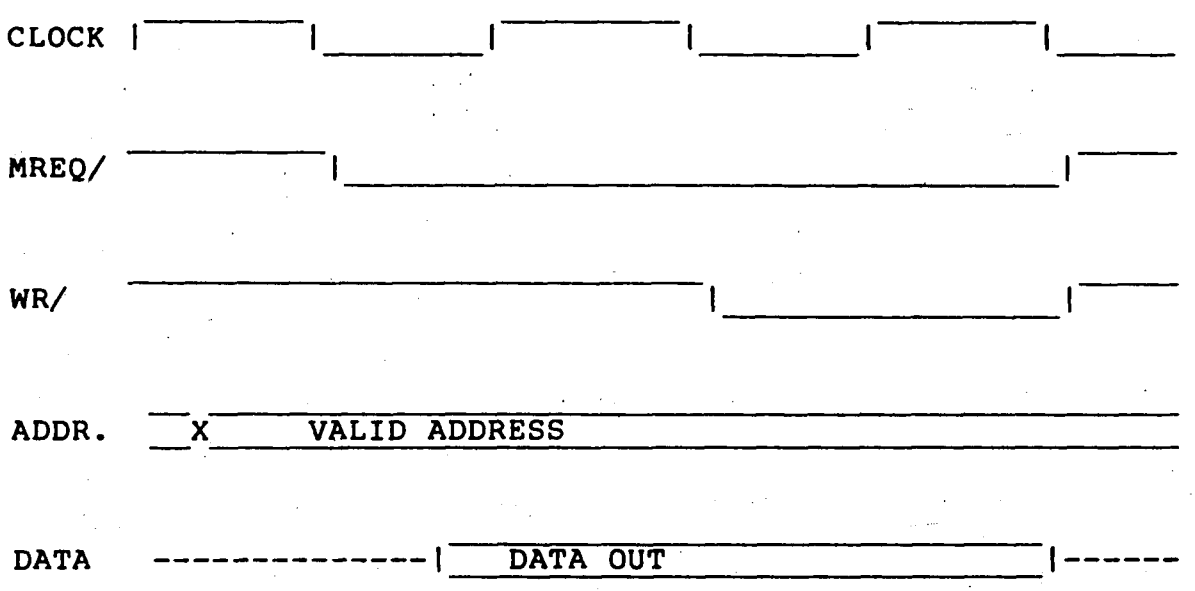

Figure IV - Memory Write Timing Diagram

The Write pulse from the Z80 appears too late in the Write cycle to be used for an Early Write (the procedure where the Write pulse is set low before Chip Enable in order to speed up the Write Cycle). In order to use the Early Write, this system generates its own Write pulse, shown as WR** in figure V. This WR** pulse is generated by Exclusive ORing MREQ/ and Read/ (which generates a high

whenever MREQ/ is low and Read/ is not) and Exclusive ORing this pulse with IORQ/ (which inverts the pulse, producing a valid Write pulse if it is not an Input or Output instruction). The result is a Write pulse that goes low as soon as MREQ/ does during a memory write cycle, but stays high any other time. The RE/, CE/ and address signals function in the same manner as in the Memory Read cycle, and all major memory transactions all occur within the first clock cycle. This clock cycle and the memory control signals are shown on the following page (figure V).

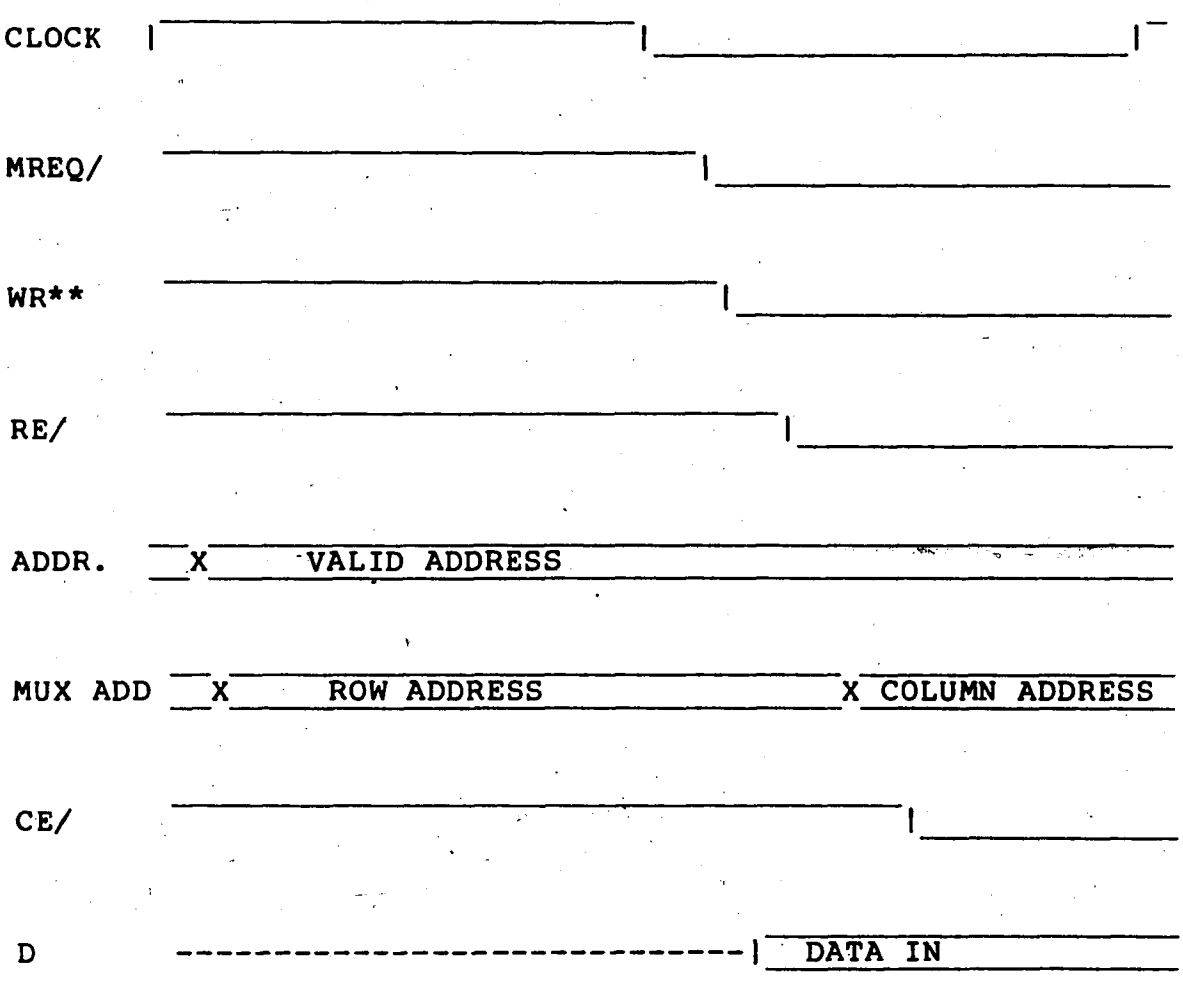

Figure V - Expanded Memory Write Timing Diagram

Normal Memory Refresh Timing

The normal memory refresh requires two clock cycles, and is done for one row after each instruction fetch. The normal refresh is controlled by chip 32 (the Intel MUX). When the

Z80 Refresh/ signal is low, chip 18 disables the CE/ input to the RAM chips by forcing it to stay high, preventing any change to the RAM data and causing the data line to remain in the high impedance state. The Refresh/ signal also causes chip 32 to place the address of the next row to be refreshed on the RAM address lines. MREQ/, which drops low half a clock cycle (.278 microseconds) after the Refresh/ signal, preforms the actual refresh of the RAM memories. When the Refresh/ signal goes high again, it triggers chip 22, a 74LS123 Retriggerable Monostable Multivibrator. Chip 22 applies a 70 ns low signal to the Count/ input of the MUX, which increments the address to be used for the next refresh by one. The timing diagram for a normal refresh cycle is shown in figure VI.

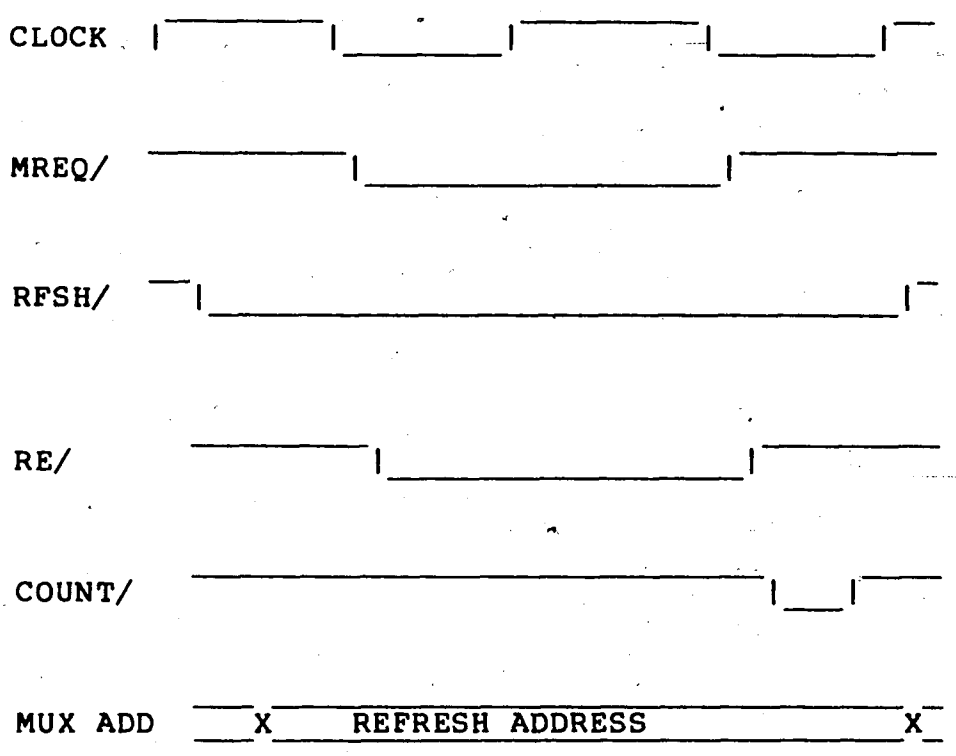

Figure VI - Normal Refresh Timing Diagram

Artificial Memory Refresh Timing

If the Z80 is placed into a Wait state for a long period of time the refreshing of the dynamic RAM chips must be taken over by the memory board itself. A wait pulse longer than 34 us during every instruction (or a single very long wait pulse) would violate the worst-case refresh conditions, and could cause a loss of memory data. In order to prevent the loss of data, the memory board contains an "Artificial

Refresh" circuit to guarantee that the refresh requirements are met. If a Wait pulse of longer than 30 us occurs the Artificial Refresh circuit will take control of the memory, temporarily disabling signals from the Z80.

When the Wait/ signal from the Z80 goes low, chip 23, a 74135 Exclusive Or/Nor gate, triggers chip 21, a 74LS123 Retriggerable Monostable Multivibrator which has been set-up to generate a 30 us pulse (referred to as Timeout in the timing diagram that follows). If the Wait/ signal returns to the high state before the end of this 30 us pulse, chip 23 raises the clear input to the 74LS123 (Chip 21), resetting the Artificial Refresh circuit. If the 30 us Timeout pulse ends with wait still low, the an artificial refresh cycle is triggered.

When the Timeout pulse goes high, it causes the second 74LS123 of chip 21 to generate a 300 ns pulse, which is sent to three places. This 300 ns pulse travels through chip 20, a 7407 Open Collector Hex Buffer Driver, and forces the Z80 Wait/ line to remain low during the artificial refresh so that the Z80 can not interfere. The 300 ns pulse goes to the Intel MUX Refresh/ input, that causes the next refresh address to be placed on the RAM address lines, and also to the RE/ input of the RAM chips, where it preforms the actual refresh of the memory. When

the 300 ns pulse ends, it causes the Count/ input of the Intel MUX to pulse low to increment the refresh address, and it restarts the 30 us Timeout pulse. If the Z80 Wait/ pulse has ended during the artificial refresh, the Timeout pulse is not restarted and control of the memory is returned to the Z80.

The timing diagram for this procedure is shown below in figure VII.

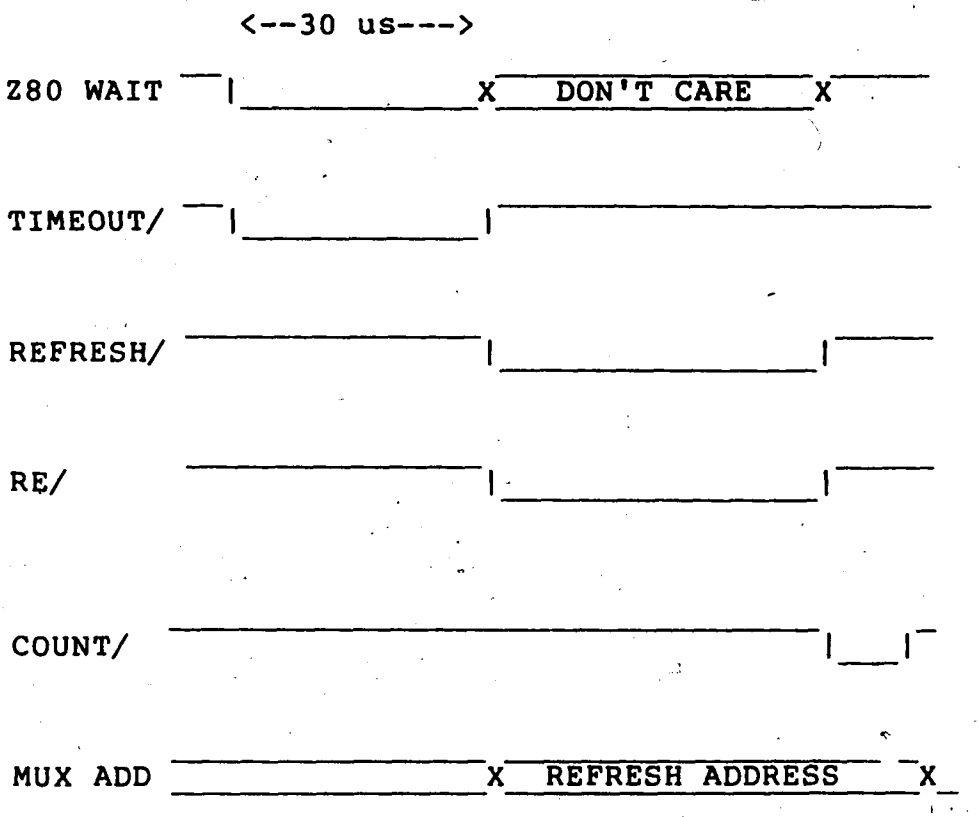

Figure VII - Artificial Refresh Timing Diagram

RAM/ROM Select

Pin 5 of chip 18, a 74S260 Dual 5-input Nor gate, determines whether the ROM or RAM memory is active. If address bits A11 to A15 are low (logic zero), pin five of chip 18 (the output of the first Nor gate) is high, which selects the ROM memory and disables the RAM. If any of these address bits are in the logic one state, the ROM will be disabled and the RAM will be active.

A logic one at the output of the first Nor gate (pin 5) disables the RAM by forcing the output of the second Nor gate (pin 6) to be a logic zero, no matter what the state of the other inputs. The output of this second Nor gate is inverted and used as the RAM memory Chip Enable/. With the Chip Enable/ high, the RAM memory can never be selected. The logic one output of this Nor gate is also inverted and passed to the Output Enable/ input of the ROM memory, allowing the ROM memory to be activated.

If any of the address bits A11 to A15 are high, the output of the second Nor gate will be under control of its other inputs (allowing the RAM to be activated when required); and the Output Enable/ of the ROM memory will be high, disabling the ROM.

II-E) SCHEMATICS AND CHIP LISTS

The following pages contain the schematic diagrams for the three hardware sections, and the chip lists that contain the information necessary to understand these schematics. Each of the integrated circuits (chips) on the diagrams are represented by a box or an appropriate symbol, with a number in the center. To determine the chip name, type, function, and power and ground connections, refer to the charts on pages 35, 36 and 37.

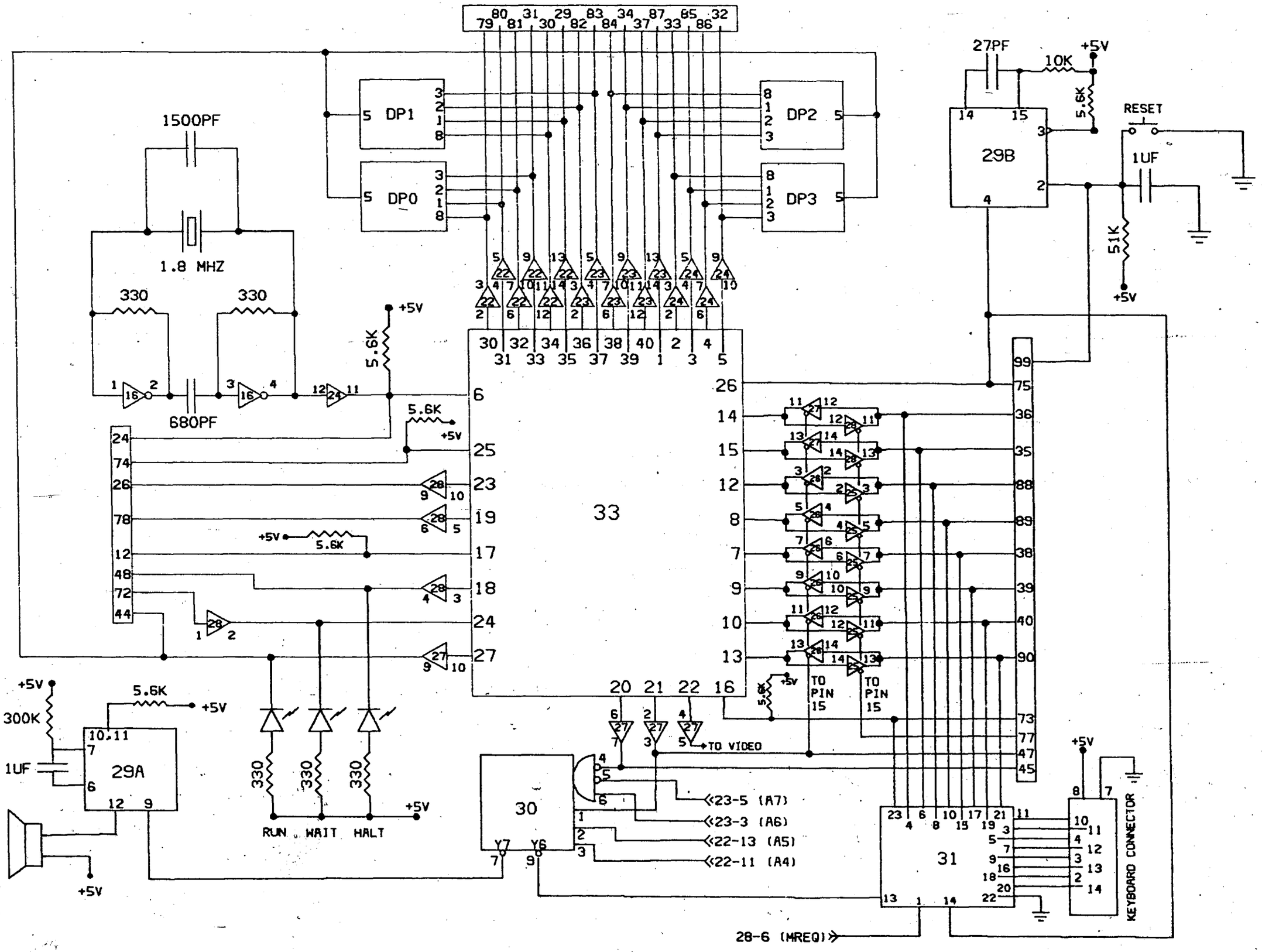

FIGURE VIII - CPU BOARD SCHEMATIC - PAGE 34A

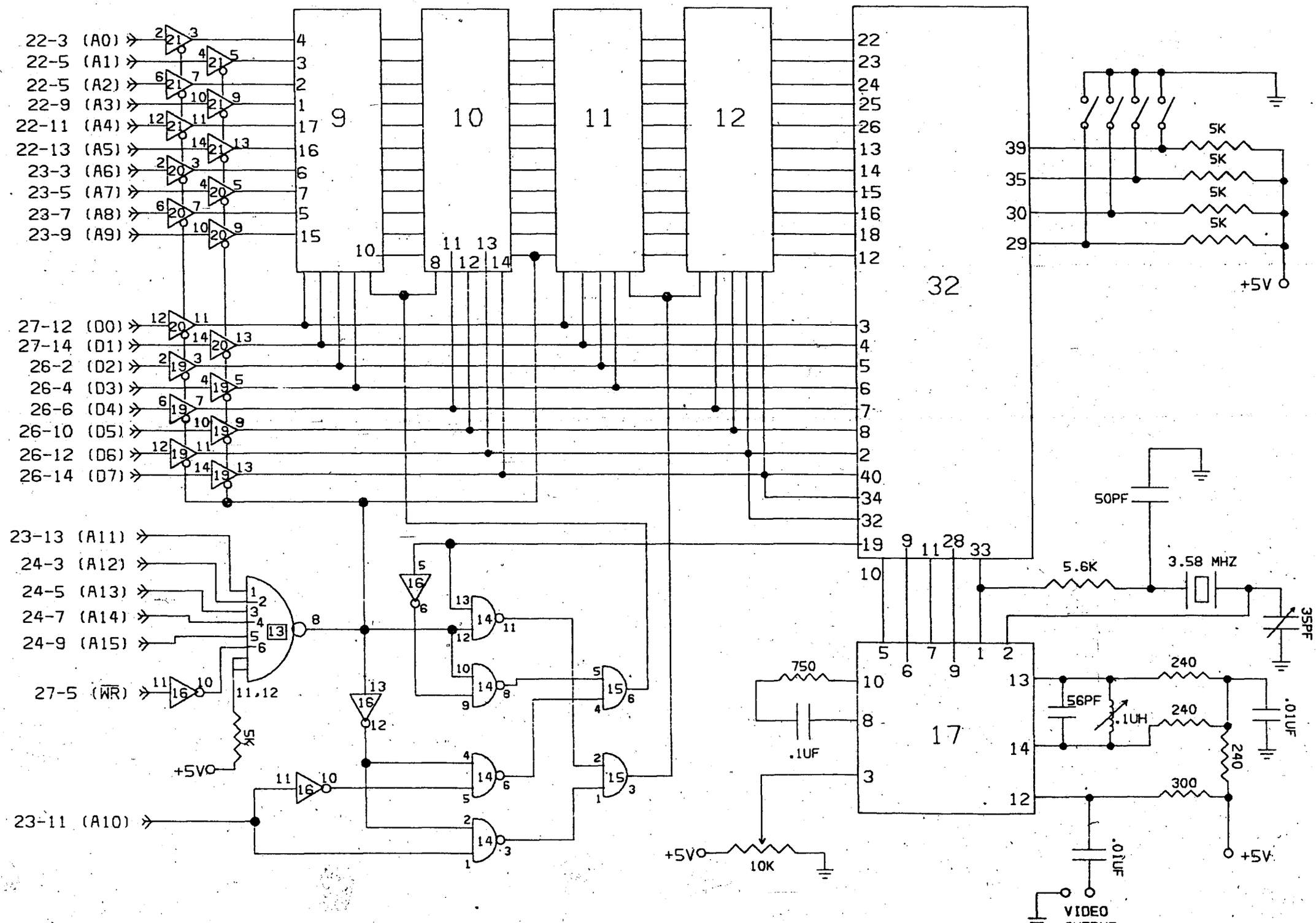

FIGURE IX - VIDEO BOARD SCHEMATIC - PAGE 34B

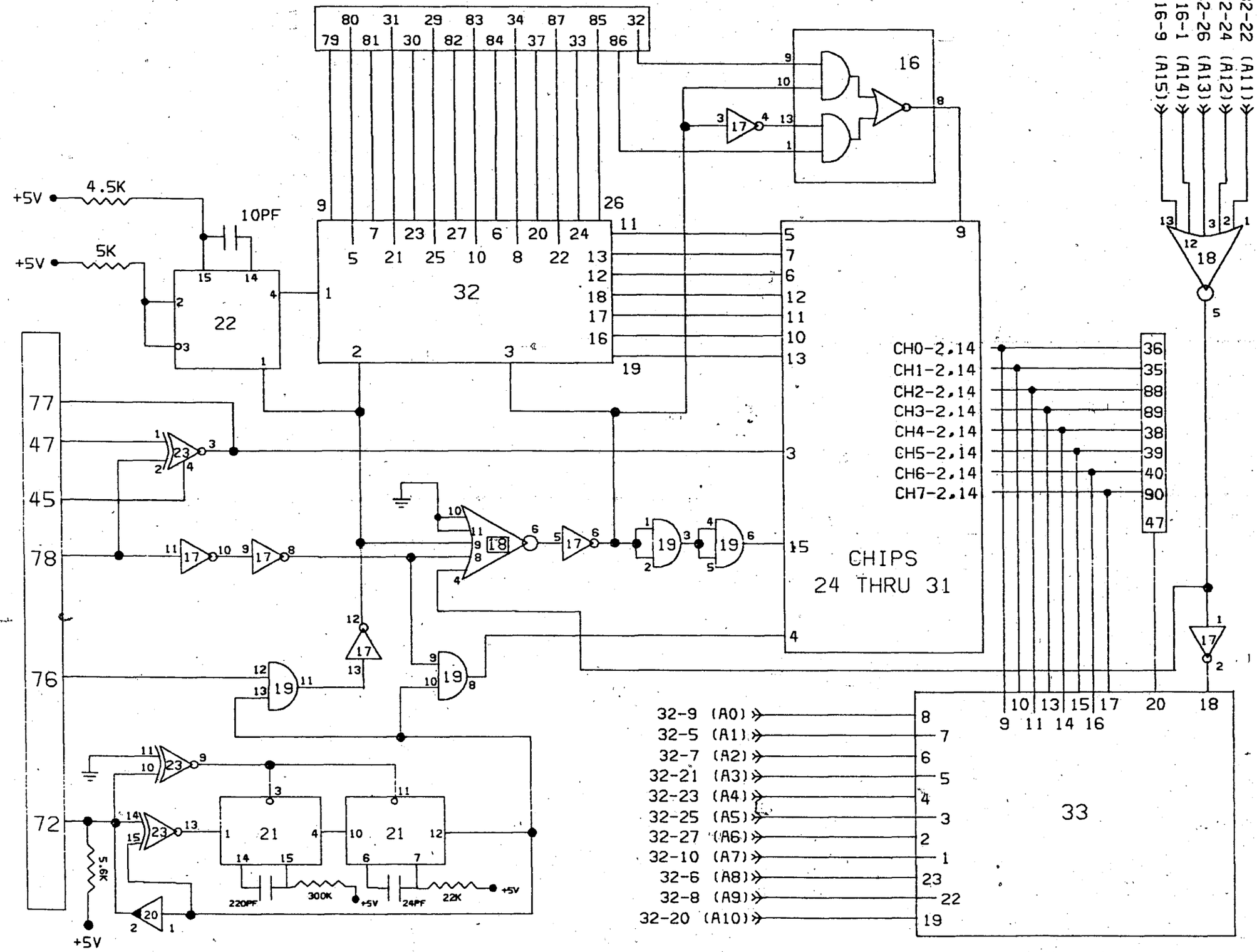

FIGURE X - MEMORY BOARD SCHEMATIC - PAGE 34C

CPU DIAGRAM KEY

Chip #	Chip Name	Chip Type	Function	Ground Conns.	5 Volt Conns.
33	Zilog Z80	CPU	Microprocessor	29	11
31	CDP1852	Decoder	Keyboard Interface	2,12	24
30	74LS138	Decoder	Output Port Decode	8	16
29	74LS123	One-Shot	Reset/Sound Timing	1,8	16
28	74LS367	Drivers	Data Bus / Control	1,8	16
27	74LS367	Drivers	Data Bus / Control	1,8	16
26	74LS365	Drivers	Data Bus	1,8	16
25	74LS365	Drivers	Data Bus	1,8	16
24	74LS365	Drivers	Address Bus	1,8,15	16
23	74LS365	Drivers	Address Bus	1,8,15	16
22	74LS365	Drivers	Address Bus	1,8,15	16
16	74LS04	Inverter	Clock/Video CS	7	14

VIDEO DIAGRAM KEY

Chip #	Chip Name	Chip Type	Function	Ground Conns.	5 Volt Conns.
32	MC6847	CPU	Video Processor	1,27,31	17
21	74LS365	Drivers	Video Address Bus	1,8	16
20	74LS365	Drivers	Vid Address / Data	1,8	16
19	74LS365	Drivers	Video Data Bus	1,8	16
17	MC1372	Modulatr	Television Display	4	11
16	74LS04	Inverter	Clock/Video CS	7	14
15	74LS08	AND Gate	Video RAM CS	7	14
14	74LS00	NAND	Video RAM CS	7	14
13	74LS30	NAND	Video RAM Select	7	14
12 THRU 09	4K SRAM	Memory	Video SRAM Memory 0 THRU Video SRAM Memory 3	9	18
DP0	5082-7395	Display	Add 0-3 Display	4,6	7
DP1	5082-7395	Display	Add 4-7 Display	4,6	7
DP2	5082-7395	Display	Add 8-11 Display	4,6	7
DP3	5082-7395	Display	Add 12-15 Display	4,6	7

MEMORY DIAGRAM KEY

Chip #	Chip Name	Chip Type	Function	Ground Conns.	5 Volt Conns.
33	2716	EPROM	ROM Memory	12	21,24
32	3242A	MUX	RAM Control/Refresh	14	28
31 THRU 24	64K RAM	Memory	DRAM Memory Chip 0 THRU DRAM Memory Chip 7	16	8
23	74LS135	EX. NOR	Artificial Refresh	4,8,11,12	16
22	74LS123	One-Shot	Refresh Count	8	16
21	74LS123	One-Shot	Artificial Refresh	8,9	2,16
20	74LS07	OC Drivr	Art. Refresh Wait	7	14
19	74LS08	AND Gate	RAM Cas and Ras	7	14
18	74S260	NOR Gate	ROM-RAM Select/Cas	7,10,11	14
17	74LS04	Inverter	Miscellaneous	7	14
16	74LS51	AND/NOR	RAM A7 Generation	7	14

III) SOFTWARE

III-A) SOFTWARE OVERVIEW

A major part of any good personal computer system is the software; a user friendly system control program containing routines that will allow the system to be used easily and effectively.

The following monitor program (microcomputer operating system) was written with these goals in mind. It was constructed exclusively for the computer system described in the hardware section in order to make the best use of all of this systems special features. The monitor program was written using the Z80 assemble language as described in the "Z80 Assemble Language Programming Manual", from Zilog.

The monitor has been divided into several sections (by function), and each of these sections is discussed in the following pages. A copy of the actual software, containing addresses, op codes, mnemonics and comments, is attached (see section III-H), and should be referred to while reading the monitor sections.

A list of all the commands accepted by the monitor is given below. These commands have been divided into four sections, according to the various functions.

DISPLAY (64 Locations)

A - Advance 64 and Display
D - Decrement 64 and Display
S - Single Advance and Display
C - Change Address and Display
F - Find Pattern and Display

MODIFY

M - Move Program from xxx to xxx

E - Edit Contents of Memory

Edit Commands are:

0 through F Hex - Write Data to RAM Memory
S - Single Advance, Memory Unchanged
X - Backup One, Memory Unchanged
N - New Address to be Edited
Q - Quit, Exit Editor

RUN

R - Run Program Beginning at First Location Displayed

TEST

POWER UP - Run Quick Self Tests
T - Run Self Tests

Figure XI - Monitor Program Command Summary

III-B) MAIN

The program section referred to as Main is the actual control program. It calls the appropriate subroutines to allow data to be input or output, and contains the code for some of the simpler commands. Main is located at hexadecimal addresses 0100 through 018F, inclusive. The subroutines called from Main are discussed in detail in the following chapter.

The Main program is executed upon successful completion of the power-up self tests. It begins by calling the subroutine "Clear Screen", which fills the television screen with blanks. Next the "Screen Write" subroutine is called (with the required information present in the HL register), and the words "ENTER ADDRESS" are displayed on the television screen. The computer is now asking the operator to enter a starting address through the ASCII keyboard. All messages from the operating system are printed in the upper right quarter of the television screen, which will be referred to as the message area from now on. See figure XII on page 46 for a sample screen (the words "ENTER COMMAND" are displayed in the message area in this figure).

Main next calls the "Get 2 Hex" subroutine twice, in order to input four hexadecimal digits from the ASCII keyboard. The "40 Out" subroutine is then called to display the contents of the 40 Hex memory locations that begin at this address on the left half of the television screen.

Main now enters the section of code which is referred to as the command loop. This section calls "Screen Write", which prints the words "ENTER COMMAND" in the message area. The subroutine "Get ASCII" (called now by Main) gets one ASCII character from the ASCII keyboard and stores it in the Accumulator. This data is then compared against all known commands and checks for a match. If a match is found Main turns control over to the appropriate program. If no match is found, Main causes "UNKNOWN COMMAND" to be displayed in the message area, beeps the pizeo-electric beeper, and restarts the command loop.

III-C) SUBROUTINES USED IN MAIN

Clear Screen

The Clear Screen subroutine is located from 0004 to 000F Hex, inclusive. This program writes a space (an ASCII 40) to memory addresses F800 through F9FF, the area where the contents of the television screen are stored.

Get ASCII

The Get ASCII subroutine is located from 0034 to 003F Hex, inclusive. This program takes advantage of the fact that a maskable interrupt received when the Z80 is in interrupt mode one will restart the Z80 at location 0038 Hex. This program places the Z80 in interrupt mode one, enables interrupts, and halts the Z80 at location 0037 Hex. The Z80 will now remain halted until an interrupt is received.

When a key from the ASCII keyboard is pressed, an interrupt is generated and the data from that key is made available to the Z80 as explained in the CPU section (Section II-B). The Z80 next does an input instruction, writing the keyboard data into its accumulator, adjusts the stack (the input data was placed on the stack, where it is not required), and returns to the calling program.

Get 2 Hex

The Get 2 Hex subroutine, located between 0010 and 0033 Hex (inclusive), will place two hexadecimal digits in the accumulator. These Hex digits are read from an ASCII keyboard, so a transformation from ASCII to Hex must be preformed. This program is called when the system is expecting hexadecimal data, otherwise the Get ASCII program is used.

Get 2 Hex begins by calling Get ASCII, which places an ASCII character into the accumulator. If the character is a number (less than 10 Hex), it is shifted left four times (causing the right four bits to be zeros) and it is stored in the H register for later use. If it is a letter, nine is added to it, then it is shifted and stored in the H register. Get ASCII is then called a second time, and again nine is added to the data if it is a letter. The top half of this second piece of data is then made into zeros, which produces the second Hex digit. This is logicly ORed with the first Hex digit (from the H register) to produce the eight bit long pair of Hex digits, which are stored in the Accumulator. Get 2 Hex then returns to the calling program, and the accumulator contains the two hexadecimal digits that were entered through the ASCII keyboard.

Screen Write

The Screen Write subroutine, located from 02B8 to 02CF Hex (inclusive), is a program that lists sixteen characters of data on two lines in the message area of the television screen. These messages have been stored in the ROM at addresses 0300 through 03AF Hex.

When the Screen Write program is called, the HL register contains the starting address of the message to be listed on the television screen. The starting address of the first line of the message area is loaded into the DE register, and eight characters are transferred from the ROM memory to the video memory (and onto the TV screen) using the Z80 LDIR instruction. The starting address of the second line of the message area is then loaded into the DE register, and the last eight characters are loaded into the video memory in the same manner.

III-D) DISPLAY COMMANDS

There are five commands (as shown in figure XI) that are used to display the contents of memory on the television screen. Each command will display 64 memory locations, four on each line of the television display. A sample of the television display after the execution of a display command is shown on the following page. The command was change location to A000 Hex and Display (A "C" followed by A000). The "Enter Command" is in the message area, which is the operating system telling the operator that the system is waiting for the next command.

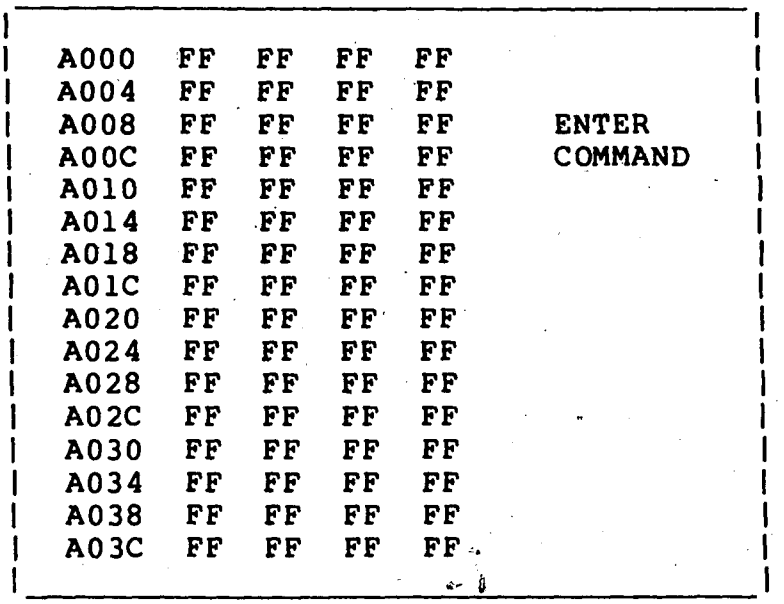

Figure XII - Sample Television Display

Display Subroutines

There are two subroutines used by the display programs. They are 40 Out, located at addresses 0040 through 0072 Hex, and 2 Out, located at addresses 0073 through 009F Hex.

The 2 Out subroutine takes the two hexadecimal digits from the accumulator, converts them to ASCII, and lists them on the television screen. Locations 0090 through 009F Hex form a table which contains the ASCII equivalents of the

last digit of the address (i.e., location 0094 contains a 34, the ASCII code for a "4"). The 2 Out subroutine loads the four hexadecimal digits "0090" into the DE register, extracts the first Hex digit (the upper four bits) from the accumulator, and logically ORs these four bits with the DE register. The DE register will then contain the address of the ASCII code for the first Hex digit. This digit is output to the television screen, and the procedure is repeated for the second Hex digit (the lower four bits).

The 40 Out subroutine lists 16 lines, each containing an address and four data words, on the television screen. This is done by first loading the two high order digits of the memory location being displayed into the accumulator and calling the 2 Out subroutine. Next the lower two address digits are output (using 2 Out), and the memory data from the address that was displayed and the three following locations is written to the screen using 2 Out. The address is incremented by four, and this procedure is repeated until all 64 memory locations to be displayed have been written to the screen. Refer to figure XII for an example of what is displayed on the television screen after the 40 Out subroutine has been called for memory location A000 (the contents of the RAM memory from A000 through A03F are all ones, displayed as Hex "FF"s).

Display Commands

The five display commands that use these subroutines are now discussed. These commands are used to examine the contents of the RAM memory by displaying 64 eight-bit words at a time on the television screen. The data is displayed in hexadecimal, in the manner explained in the previous section on the "40 Out" subroutine.

The code for the A,C,D and S commands is contained within the command loop. The code for the F command is located from address 021A to 023F Hex, inclusive.

A - Advance 64 and Display

When the system is first powered up, the user enters an address, and the 64 consecutive memory locations starting at that address are displayed. To examine memory locations which appear after this in the memory, the A command can be used. This command adds 40 Hex to the BC register (the register used as a pointer by the Z80 to tell it what location it should be addressing), and calls the 40 Out subroutine. This displays the first 64 memory locations which were not previously on the screen. The Z80 will then wait for the next command.

D - Decrement 64 and Display

The "D" command works like the "A" command, except that 40 Hex is subtracted from the BC register, and 40 Out then displays the 64 locations before the first address that was on the screen. If the address tries to go below zero, the program will wrap around through the end of RAM memory (FFFF Hex).

S - Single Advance and Display

The "S" command is the same as the "A" command, except that the BC register is only incremented by one location.

C - Change Location and Display

The "C" command allows the user to easily examine an address which is not located close to the address currently being displayed. This command causes the Z80 to begin executing the Main program again by jumping to location 0103 Hex. This will clear the screen and ask for an address. When an address is entered, that memory location and the following 63 are displayed using 40 Out. The Z80 then restarts the "Command Loop", as explained in the "MAIN" section.

F - Find Pattern and Display

The "F" command will ask for a two-digit hexadecimal pattern to search for, and then will execute a "C" command

to the first memory address where that data occurs. If the pattern does not exist anywhere in the RAM memory, "NO MATCH FOUND" will be printed in the message area, and the Z80 will wait for the next command.

The Find command first calls Get 2 Hex, which allows the eight bit "target" pattern to be input from the ASCII keyboard, and places it into the accumulator. The Z80 then sets up the registers as required and executes the CPIR instruction, which checks each memory location until a match is found or the entire memory has been checked. The system will then execute a "C" command or print "NO MATCH FOUND", has explained in the previous paragraph.

III-E) MODIFY COMMANDS

The Modify commands, Move and Edit, are used to change the contents of the RAM memory. These commands are the only way for the user to enter or modify data in the RAM memory.

M - Move

The "M" command is used to copy a section of code from one location in memory to another. The code for this command is located from address 0240 through address 027F Hex.

When the "M" command is selected, the system first displays "STARTING ADDRESS?" in the message area, and uses Get 2 Hex to input the four digit address (the beginning address of the data to be moved). The system next displays "ENDING ADDRESS?", and uses Get 2 Hex to input the end address of the data to be moved (which is placed into the HL register). The system then displays "NEW START?" and uses Get 2 Hex to input the address that will be the new beginning address of the data being moved.

The system then calculates the number of words to be moved (which is placed into register BC) and the new ending address (placed into register DE) from the information input by the user. The LDDR instruction is then used to

transfer data from memory location HL to memory location DE, decrement HL, DE and BC until BC is zero. When the LDDR instruction has finished, the move is complete. The system then beeps, uses 40 Out to display the contents of the first 64 new addresses, and asks for the next command.

E - Edit

The Edit program is used to interactively modify any of the RAM memory locations. When the Edit program is entered (by using the "E" command) it activates several new commands that are valid only in the Edit mode. These commands are listed below, and are explained in the following sections. The code for the Edit program is located from 0190 to 0219 Hex, inclusive.

0 through F Hex - Write Data to RAM Memory

S - Single Advance, Memory Unchanged

X - Backup One, Memory Unchanged

N - New Address to be Edited

Q - Quit, Exit Editor

The Edit program begins by displaying the word "EDIT" in inverse video in the bottom right quarter of the television screen, and displaying "ENTER ADDRESS" in the message area. The system then uses Get 2 Hex to input the address to be

edited. The system then uses 40 Out to display the location entered (and the following 63), prints the address being edited in inverse video under the word EDIT, and prints "ENTER DATA" in the message area. A picture of what the television screen would look like after the execution of an "E A000" is shown in figure XIII.

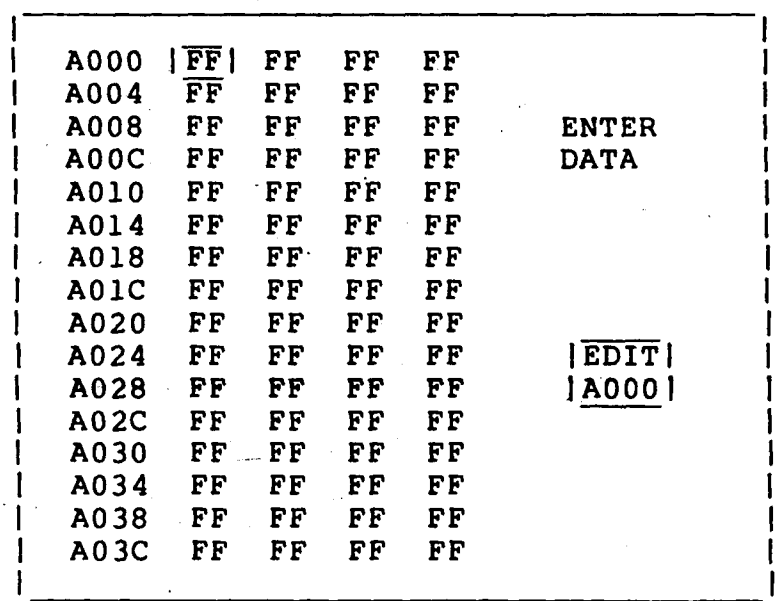

Figure XIII - Sample Edit Screen

As depicted above (by enclosure in a box), the actual eight bits of data about to be edited are displayed in inverse video. When two hexadecimal digits are entered from the keyboard they will replace the data shown in inverse video, the address displayed under the word EDIT will be

incremented by one, and the data located at this new address will be shown in inverse video.

S - Single Advance, Memory Unchanged

The "S" command can be used to skip ahead without changing the contents of the memory. This command advances the address to be edited, increments the addresses on the screen, and places the new location to be edited in inverse video.

X - Backup One, Memory Unchanged

The "X" command is used to move back one memory location. It can be used to correct mistakes or examine the contents of memory that appear before the address being edited. This command decrements the address to be edited and the address shown on the screen, and places the new location to be edited in inverse video.

N - New Address to be Edited

The "N" command will restart the Edit session. The Edit program restarts by asking for the memory address to be edited, then displays the new address to be edited (as explained in a previous section).

Q - Quit, Exit Editor

The "Q" command is used to return control to the "Main" program. This command ends the edit session, erases the word EDIT and the edit address that appears below it, and jumps to location 0100 Hex (the start of Main).

III-F) RUN COMMAND

The code for the "R" command is contained in the command loop, starting at 0143 Hex. This command will cause the Z80 to exit from the monitor program and run from instructions stored in the systems memory. When an "R" command is entered the Z80 will print the words "RUNNING PROGRAM" in the message area, and will begin executing the code that is displayed on the television screen. For example, if a C 7777 is entered (causing the system to display memory locations 7777 through 77A6 Hex) followed by an R, the system would begin executing the code located at 7777 Hex.

III-G) TEST COMMANDS

The system has two sets of self tests built into it. There is a shorter self test program, called "Power Up Check", that does a quick memory test whenever the system is reset or powered-up. There is also a Self Test program which can be run by entering the "T" command. This program runs a detailed test on the Z80, Dynamic and Static RAM memories, video display, and the ASCII keyboard.

POWER UP - Quick Self Test

The Power-Up self test program is located from address 0280 to 02B7 Hex, inclusive. This program first sets the stack pointer to FFFF, then writes an AA (10101010 in binary) to location FFFF, and a 55 (01010101 in binary) to location FFFE. The contents of these two locations are then checked, and if both contain the correct data the test passes and the Z80 begins executing the "Main" program. If either location fails, the television screen is turned red (by filling it with red graphics characters), the word stack is printed out, the system beeps the pizeo-electric buzzer, and the Z80 halts. This indicates that the Z80 cannot run any of its programs, since it cannot create the stack required for any subroutine calls.

T - Run Self Tests

The Self Test program is located at addresses 00A0 through 00FF Hex, and is run by entering the "T" command. The words "SELF TESTS" are displayed in the message area and the entire RAM memory is filled with zeros (filling the television screen with '@'s). The CPU then reads (a 00 Hex if the memory is good) and writes an FF Hex once for every RAM address (filling the television screen with orange graphic characters). Finally, the CPU again reads every RAM address, expecting to find an FF Hex. If all memory locations are correct, the self tests proceed to the keyboard test, which is described in the next paragraph. If at any time a RAM location fails, the words "RAM FAILURE", followed by the address of the location that failed, are printed in the message area, the test is stopped, and the buzzer continuously beeps.

When the RAM test has passed, the words "KEYBOARD TEST" are printed in the message area and the test for the ASCII keyboard is begun. When any key on the keyboard is pressed, the appropriate character is printed on the television screen.

III-H) MONITOR PROGRAM

The following pages contain the listing for the monitor program. This listing contains the actual code, mnemonics and comments required for a detailed discussion of the monitor program. It should be used in connection with the previous sections for a complete understanding of the programs.

ADD.	MNEMONIC	OP CODE	COMMENTS	PROGRAM NAME
0000	JP	C3	JUMP TO POWER UP CHECK	
0001		80	(MAKE SURE STACK	
0002		02	MEMORY IS OK)	
0003	NOP	00		
0004	LD,HL,nn	21	HL= START OF SCREEN ADDRESS	
0005		00		
0006		F8		CLEAR
0007	LD,(HL),n	36	WRITE A SPACE TO THE SCREEN	SCREEN
0008		20		******
0009	INC HL	23	HL= NEXT SCREEN ADDRESS	
000A	CB,7,H	CB	CHECK IF SCREEN ALL CLEARED	
000B		7C		
000C	JP,NZ	C2	JUMP IF NOT ALL CLEARED	
000D		07		
000E		00		
000F	RETURN	C9	RETURN (SCREEN ALL CLEARED)	
0010	CALL	CD	CALL GET ASCII (HIGH HEX)	
0011		34		
0012		00		GET 2
0013	CP,A,n	FE	CHECK IF NUMBER OR LETTER	HEX

```
0014|           | 10 |                                  |  *****
0015| JP,NC     | D2 | JUMP IF NUMBER                   |
0016|           | 1A |                                  |
0017|           | 00 |                                  |
0018| ADD,A,n   | C6 | ADD 9 (TURN ASCII TO HEX)        |
0019|           | 09 |                                  |
001A| RLCA      | 07 | LEFT JUSTIFY HIGH HEX DIGIT      |
001B| RLCA      | 07 |                                  |
001C| RLCA      | 07 |                                  |
001D| RLCA      | 07 |                                  |
001E| AND,A,n   | E6 | REMOVE LOW HALF (IT'S JUNK)      |
001F|           | F0 |                                  |
0020| LD,H,A    | 67 | STORE HIGH HEX DIGIT             |
0021| CALL      | CD | CALL GET ASCII (LOW HEX)         |
0022|           | 34 |                                  |
0023|           | 00 |                                  |
0024| CP,A,n    | FE | CHECK IF NUMBER OR LETTER        |
0025|           | 10 |                                  |
0026| JP,NC     | D2 | JUMP IF LETTER                   |
0027|           | 2B |                                  |
0028|           | 00 |                                  |
0029| ADD,A,n   | C6 | ADD 9 (TURN ASCII TO HEX)        |
002A|           | 09 |                                  |
002B| AND,A,n   | E6 | REMOVE TOP HALF (IT'S JUNK)      |
002C|           | 0F |                                  |
002D| OR,A,H    | B4 | COMBINE HIGH AND LOW HEX         |
002E| RETURN    | C9 | RETURN (A CONTAINS RESULT)       |
002F| NOP       | 00 |                                  |
0030| NOP       | 00 |                                  |
0031| NOP       | 00 |                                  |
0032| NOP       | 00 |                                  |
0033| NOP       | 00 |                                  |
----+-----------+----+----------------------------------+--------
0034| IM 1      | ED | SET INTERRUPT MODE 1             |
0035|           | 56 |                                  |
0036| EI        | FB |                                  |
0037| HALT      | 76 | WAIT FOR INPUT (INTERRUPT)       |   GET
0038| IN,A,(n)  | DB | READ DATA FROM PORT 7F           |  ASCII
0039|           | 7F |                                  |  *****
003A| DI        | F3 |                                  |
003B| INC SP    | 33 | REMOVE OLD PC FROM STACK         |
003C| INC SP    | 33 |                                  |
003D| RETURN    | C9 | RETURN (A CONTAINS DATA)         |
003E| NOP       | 00 |                                  |
003F| NOP       | 00 |                                  |
----+-----------+----+----------------------------------+--------
```

```
----+----------+----+-------------------------------+--------
0040| LD,HL,nn | 21 | HL= START OF SCREEN ADDRESS   |
0041|          | 00 |                               |
0042|          | F8 |                               |  40 OUT
0043| PUSH BC  | C5 | SAVE CURRENT ADDRESS          |  ******
0044| LD,A,B   | 78 |                               |
0045| CALL     | CD | CALL 2 OUT                    |
0046|          | 73 |  WRITE HIGH HALF OF DISPLAY   |
0047|          | 00 |   ADDRESS TO SCREEN           |
0048| DEC HL   | 2B | ADJUST SCREEN ADDRESS         |
0049| DEC HL   | 2B |                               |
004A| LD,A,C   | 79 |                               |
004B| CALL     | CD | CALL 2 OUT                    |
004C|          | 73 |   WRITE LOW HALF OF DISPLAY   |
004D|          | 00 |   ADDRESS TO SCREEN           |
004E| LD,A,(BC)| 0A |                               |
004F| CALL     | CD | CALL 2 OUT                    |
0050|          | 73 |   WRITE MEMORY CONTENTS       |
0051|          | 00 |   TO SCREEN                   |
0052| INC BC   | 03 |                               |
0053| LD,A,(BC)| 0A |                               |
0054| CALL     | CD | CALL 2 OUT                    |
0055|          | 73 |   WRITE MEMORY CONTENTS       |
0056|          | 00 |   TO SCREEN                   |
0057| INC BC   | 03 |                               |
0058| LD,A,(BC)| 0A |                               |
0059| CALL     | CD | CALL 2 OUT                    |
005A|          | 73 |   WRITE MEMORY CONTENTS       |
005B|          | 00 |   TO SCREEN                   |
005C| INC BC   | 03 |                               |
005D| LD,A,(BC)| 0A |                               |
005E| CALL     | CD | CALL 2 OUT                    |
005F|          | 73 |   WRITE MEMORY CONTENTS       |
0060|          | 00 |   TO SCREEN                   |
0061| INC BC   | 03 |                               |
0062| LD,A,L   | 7D |                               |
0063| ADD,A,n  | C6 | ADD 0A TO HL (SET ADDRESS     |
0064|          | 0A |   TO NEXT DISPLAY LOCATION)   |
0065| LD,L,A   | 6F |                               |
0066| LD,A,H   | 7C |                               |
0067| ADC,A,n  | CE |                               |
0068|          | 00 |                               |
0069| LD,H,A   | 67 |                               |
006A| CP,A,n   | FE | CHECK IF ALL 40 LOCATIONS     |
006B|          | FA |   HAVE BEEN DISPLAYED         |
006C| JP,NZ    | C2 | JUMP IF NOT DONE              |
006D|          | 44 |                               |
006E|          | 00 |                               |
006F| POP BC   | C1 | RESTORE CURRENT ADDRESS       |
```

```
0070| RETURN    | C9 | RETURN                          |
0071| NOP       | 00 |                                 |
0072| NOP       | 00 |                                 |
----+-----------+----+---------------------------------+--------
0073| LD,D,n    | 16 | D= HIGH HALF OF LOOK-UP         |
0074|           | 00 |     TABLE ADDRESS               |
0075| PUSH AF   | F5 | SAVE 2OND HEX TILL LATER        |  2 OUT
0076| RRCA      | 0F | RIGHT JUSTIFY FIRST HEX         |  *****
0077| RRCA      | 0F |                                 |
0078| RRCA      | 0F |                                 |
0079| RRCA      | 0F |                                 |
007A| AND,A,n   | E6 | GET FIRST HEX DIGIT ONLY        |
007B|           | 0F |                                 |
007C| OR,A,n    | F6 | TURN DIGIT INTO ADDRESS         |
007D|           | 90 |                                 |
007E| LD,E,A    | 5F | DE= LOOK-UP TABLE ADDRESS       |
007F| LD,A,(DE) | 1A | GET ASCII EQUIVALENT            |
0080| LD,(HL),A | 77 | WRITE IT TO SCREEN              |
0081| INC HL    | 23 |                                 |
0082| POP AF    | F1 | RESTORE HEX PAIR                |
0083| AND,A,n   | E6 | GET SECOND HEX DIGIT            |
0084|           | 0F |                                 |
0085| OR,A,n    | F6 | TURN DIGIT INTO ADDRESS         |
0086|           | 90 |                                 |
0087| LD,E,A    | 5F | DE= LOOK-UP TABLE ADDRESS       |
0088| LD,A,(DE) | 1A | GET ASCII EQUIVALENT            |
0089| LD,(HL),A | 77 | WRITE IT TO SCREEN              |
008A| INC HL    | 23 |                                 |
008B| INC HL    | 23 | ADD TWO SPACES                  |
008C| INC HL    | 23 |                                 |
008D| RETURN    | C9 | RETURN                          |
008E| NOP       | 00 |                                 |
008F| NOP       | 00 |                                 |
----+-----------+----+---------------------------------+--------
0090| "0"       | 30 | EACH LOCATION CONTAINS THE      |
0091| "1"       | 31 | ASCII EQUIVALENT FOR THE        |
0092| "2"       | 32 | LAST HEX DIGIT OF ITS           |  2 OUT
0093| "3"       | 33 | ADDRESS                         |  LOOK-UP
0094| "4"       | 34 |                                 |  TABLE
0095| "5"       | 35 |                                 |  *******
0096| "6"       | 36 |                                 |
0097| "7"       | 37 |                                 |
0098| "8"       | 38 |                                 |
0099| "9"       | 39 |                                 |
009A| "A"       | 01 |                                 |
009B| "B"       | 02 |                                 |
009C| "C"       | 03 |                                 |
009D| "D"       | 04 |                                 |
009E| "E"       | 05 |                                 |
```

```
009F| "F"        | 06 |                                  |
----+-----------+----+----------------------------------+--------
00A0| LD,HL,nn  | 21 | HL= ADDRESS OF "SELF TESTS"      |
00A1|           | 30 |                                  |
00A2|           | 03 |                                  | SELF
00A3| CALL      | CD | CALL SCREEN WRITE                | TESTS
00A4|           | B8 |                                  | *****
00A5|           | 02 |                                  |
00A6| LD,HL,nn  | 21 | HL= START OF RAM MEMORY          |
00A7|           | 00 |                                  |
00A8|           | 08 |                                  |
00A9| XOR,A,A   | AF | A=0                              |
00AA| LD,(HL),A | 77 | LOAD "00" INTO MEMORY            |
00AC| INC HL    | 23 |                                  |
00AD| CP,A,H    | BC | CHECK IF ALL LOADED              |
00AE| JP,NZ     | C2 | JUMP IF NOT ALL LOADED           |
00AF|           | AA |                                  |
00B0|           | 00 |                                  |
00B1| LD,H,n    | 26 | HL= START OF RAM MEMORY          |
00B2|           | 08 |                                  |
00B3| LD,A,(HL) | 7E | GET MEMORY DATA                  |
00B4| CP,A,n    | FE | CHECK IF DATA = "00"             |
00B5|           | 00 |                                  |
00B6| JP,NZ     | C2 | JUMP IF ITS NOT (ITS BAD)        |
00B7|           | E0 |   TO RAM TEST FAILURE            |
00B8|           | 00 |                                  |
00B9| XOR,A,A   | AF | A=0                              |
00BA| LD,(HL),n | 36 | LOAD "FF" INTO MEMORY            |
00BC|           | FF |                                  |
00BB| INC HL    | 23 |                                  |
00BC| CP,A,H    | BC | CHECK IF ALL MEMORY TESTED       |
00BD| JP,NZ     | C2 | JUMP IF NOT ALL TESTED           |
00BE|           | B3 |                                  |
00BF|           | 00 |                                  |
00C0| LD,H,n    | 26 | HL= START OF RAM MEMORY          |
00C1|           | 08 |                                  |
00C2| LD,A,(HL) | 7E | GET MEMORY DATA                  |
00C3| CP,A,n    | FE | CHECK IF DATA = "FF"             |
00C4|           | FF |                                  |
00C5| JP,NZ     | C2 | JUMP IF ITS NOT (ITS BAD)        |
00C6|           | E0 |   TO RAM TEST FAILURE            |
00C7|           | 00 |                                  |
00C8| INC HL    | 23 |                                  |
00C9| CP,A,H    | BC | CHECK IF ALL MEMORY TESTED       |
00CA| JP,NZ     | C2 | JUMP IF NOT ALL TESTED           |
00CB|           | C2 |                                  |
00CC|           | 00 |                                  |
00CD| OUT,n,A   | D3 | WRITE A "BEEP"                   |
00CE|           | 7F |   (PORT 7F)                      |
```

```
00CF| LD,HL,nn  | 21 | ADDRESS OF "KEYBOARD TEST"   |
00D0|           | 40 |                              |
00D1|           | 03 |                              |
00D2| CALL      | CD | CALL SCREEN WRITE            |
00D3|           | B8 |                              |
00D4|           | 02 |                              |
00D5| LD,HL,nn  | 21 | HL= START OF SCREEN ADDRESS  |
00D6|           | 00 |                              |
00D7|           | F8 |                              |
00D8| CALL      | CD | CALL GET ASCII               |
00D9|           | 34 |   GET DATA FROM KEYBOARD     |
00DA|           | 00 |                              |
00DB| LD,(HL),A | 77 | WRITE DATA TO TV SCREEN      |
00DC| INC HL    | 23 |                              |
00DD| JP        | C3 | DO IT AGAIN (UNTIL RESET)    |
00DE|           | D8 |                              |
00DF|           | 00 |                              |
----+-----------+----+------------------------------+--------
00E0| OUT,n,A   | D3 | WRITE A "BEEP"               |
00E1|           | 7F |   (PORT 7F)                  |  RAM
00E2| LD,A,H    | 7C | SAVE H                       |  TEST
00E3| LD,B,L    | 45 | SAVE L                       | FAILURE
00E4| LD,HL,nn  | 21 | ADDRESS OF "RAM FAILURE"     | *******
00E5|           | 50 |                              |
00E6|           | 03 |                              |
00E7| CALL      | CD | CALL SCREEN WRITE            |
00E8|           | B8 |                              |
00E9|           | 02 |                              |
00EA| LD,HL,nn  | 21 | ADDRESS OF 3RD DISPLAY LINE  |
00EB|           | 97 |                              |
00EC|           | F8 |                              |
00ED| CALL      | CD | CALL 2 OUT                   |
00EE|           | 73 |   WRITE HIGH ADDRESS OF      |
00EF|           | 00 |   FAILED RAM LOCATION        |
00F0| DEC HL    | 2B | ADJUST SCREEN ADDRESS        |
00F1| DEC HL    | 2B |                              |
00F2| LD,A,B    | 78 |                              |
00F3| CALL      | CD | CALL 2 OUT                   |
00F4|           | 73 |   WRITE LOW ADDRESS OF       |
00F5|           | 00 |   FAILED RAM LOCATION        |
00F6| OUT,A,n   | D3 | WRITE A "BEEP"               |
00F7|           | 7F |                              |
00F8| JP        | C3 | REPEAT (MEMORY NEEDS FIXED)  |
00F9|           | F6 |                              |
00FA|           | 00 |                              |
00FB| NOP       | 00 |                              |
00FC| NOP       | 00 |                              |
00FD| NOP       | 00 |                              |
00FE| NOP       | 00 |                              |
```

```
00FF| NOP      | 00 |                                    |
----+---------+----+------------------------------------+--------
0100| NOP      | 00 |                                    |
0101| NOP      | 00 |                                    | ****
0102| NOP      | 00 |                                    | MAIN
0103| CALL     | CD | CALL CLEAR SCREEN                  | ****
0104|          | 04 |                                    |
0105|          | 00 |                                    |
0106| LD,HL,nn | 21 | ADDRESS OF "ENTER ADDRESS"         |
0107|          | 00 |                                    |
0108|          | 03 |                                    |
0109| CALL     | CD | CALL SCREEN WRITE                  |
010A|          | B8 |                                    |
010B|          | 02 |                                    |
010C| CALL     | CD | CALL GET 2 HEX                     |
010D|          | 10 |   GET HIGH ADDRESS                 |
010E|          | 00 |                                    |
010F| LD,B,A   | 47 |                                    |
0110| CALL     | CD | CALL GET 2 HEX                     |
0111|          | 10 |   GET LOW ADDRESS                  |
0112|          | 00 |                                    |
0113| LD,C,A   | 4F | BC= ADDRESS TO BE DISPLAYED        |
0114| CALL     | CD | CALL 40 OUT                        |
0115|          | 40 |                                    |
0116|          | 00 |                                    |
0117| NOP      | 00 |*** START OF COMMAND LOOP ***       |
0118| LD,HL,nn | 21 | ADDRESS OF "ENTER COMMAND"         |
0119|          | 10 |                                    |
011A|          | 03 |                                    |
011B| CALL     | CD | CALL SCREEN WRITE                  |
011C|          | B8 |                                    |
011D|          | 02 |                                    |
011E| CALL     | CD | CALL GET ASCII                     |
011F|          | 34 |   GET COMMAND OR DATA              |
0120|          | 00 |                                    |
0121| CP,A,n   | FE | CHECK IF COMMAND = ADVANCE         |
0122|          | 01 |                                    |
0123| JP,NZ    | C2 | JUMP IF NOT ADVANCE                |
0124|          | 32 |                                    |
0125|          | 01 |                                    |
0126| LD,A,C   | 79 | BC = BC + 40                       |
0127| ADD,A,n  | C6 |                                    |
0128|          | 40 |                                    |
0129| LD,C,A   | 4F |                                    |
012A| LD,A,B   | 78 |                                    |
012B| ADC,A,n  | CE |                                    |
012C|          | 00 |                                    |
012D| LD,B,A   | 47 |                                    |
012E| JP       | C3 | JUMP TO CALL 40 OUT                |
```

```
012F|           | 14 |
0130|           | 01 |
0131| NOP       | 00 |
0132| CP,A,n    | FE | CHECK IF CMD = DECREMENT
0133|           | 04 |
0134| JP,NZ     | C2 | JUMP IF NOT DECREMENT
0135|           | 43 |
0136|           | 01 |
0137| LD,A,C    | 79 | BC = BC - 40
0138| SUB,A,n   | D6 |
0139|           | 40 |
013A| LD,C,A    | 4F |
013B| LD,A,B    | 78 |
013C| SBB,A,n   | DE |
013D|           | 00 |
013E| LD,B,A    | 47 |
013F| JP        | C3 | JUMP TO CALL 40 OUT
0140|           | 14 |
0141|           | 01 |
0142| NOP       | 00 |
0143| CP,A,n    | FE | CHECK IF COMMAND = RUN
0144|           | 12 |
0145| JP,NZ     | C2 | JUMP IF NOT RUN
0146|           | 52 |
0147|           | 01 |
0148| LD,HL,nn  | 21 | HL= ADDRESS OF
0149|           | 20 |    "RUNNING PROGRAM"
014A|           | 03 |
014B| CALL      | CD | CALL SCREEN WRITE
014C|           | B8 |
014D|           | 02 |
014E| LD,H,B    | 60 | HL = ADDRESS OF PROGRAM
014F| LD,L,C    | 69 |
0150| JP (HL)   | E9 | JUMP TO PROGRAM START
0151| NOP       | 00 |
0152| CP,A,n    | FE | CHECK IF CMD = CHANGE ADD.
0153|           | 03 |
0154| JP,Z      | CA | JUMP IF CHANGE ADDRESS
0155|           | 03 |
0156|           | 01 |
0157| NOP       | 00 |
0158| CP,A,n    | FE | CHECK IF COMMAND = EDIT
0159|           | 05 |
015A| JP,Z      | CA | JUMP IF EDIT
015B|           | 90 |
015C|           | 01 |
015D| NOP       | 00 |
015E| CP,A,n    | FE | CHECK IF COMMAND = TEST
015F|           | 14 |
```

0160	JP,Z	CA	JUMP IF TEST (SELF TESTS)
0161		A0	
0162		00	
0163	NOP	00	
0164	CP,A,n	FE	CHECK IF COMMAND = FIND
0165		06	
0166	JP,Z	CA	JUMP IF FIND
0167		1A	
0168		02	
0169	NOP	00	
016A	CP,A,n	FE	CHECK IF COMMAND = MOVE
016B		0D	
016C	JP,Z	CA	JUMP IF MOVE
016D		40	
016E		02	
0174	CP,A,n	FE	CHECK IF CMD = SNGL ADVANCE
0175		13	
0176	JP,NZ	C2	JUMP IF NOT SINGLE ADVANCE
0177		78	
0178		01	
0179	INC BC	03	BC = BC + 1
017A	JP	C3	JUMP TO CALL 40 OUT
017B		14	
0177		01	
0178	OUT,n,A	D3	WRITE A "BEEP"
0179		7F	(UNKNOWN COMMAND)
017A	JP	C3	JUMP TO START OF CMD LOOP
017B		18	
017C		01	
017D	NOP	00	
017E	NOP	00	
017F	NOP	00	RESERVED FOR COMMAND
0180	NOP	00	LOOP EXPANSION
0181	NOP	00	
0182	NOP	00	
0183	NOP	00	
0184	NOP	00	
0185	NOP	00	
0186	NOP	00	
0187	NOP	00	
0188	NOP	00	
0189	NOP	00	
018A	NOP	00	
018B	NOP	00	
018C	NOP	00	
018D	NOP	00	
018E	NOP	00	
018F	NOP	00	

```
----+----------+----+-------------------------------+--------
0190| LD,HL,nn | 21 | HL= 4TH TV OUTPUT LINE        |
0191|          | 17 |                               |
0192|          | F9 |                               |
0193| NOP      | 00 |                               |  EDIT
0194| LD,(HL),n| 36 | WRITE "E" TO SCREEN           |  ****
0195|          | 05 |                               |
0196| INC HL   | 23 |                               |
0197| LD,(HL),n| 36 | WRITE "D" TO SCREEN           |
0198|          | 04 |                               |
0199| INC HL   | 23 |                               |
019A| LD,(HL),n| 36 | WRITE "I" TO SCREEN           |
019B|          | 09 |                               |
019C| INC HL   | 23 |                               |
019D| LD,(HL),n| 36 | WRITE "T" TO SCREEN           |
019E|          | 14 |                               |
019F| LD,HL,nn | 21 | HL= ADDR OF "ENTER ADDRESS"   |
01A0|          | 00 |                               |
01A1|          | 03 |                               |
01A2| CALL     | CD | CALL SCREEN WRITE             |
01A3|          | B8 |                               |
01A4|          | 02 |                               |
01A5| CALL     | CD | CALL GET 2 HEX                |
01A6|          | 10 |   GET 2 HIGH ADDRESS DIGITS   |
01A7|          | 00 |                               |
01A8| LD,B,A   | 47 | BC = ADDRESS TO BE EDITED     |
01A9| CALL     | CD | CALL GET 2 HEX                |
01AA|          | 10 |   GET 2 LOW ADDRESS DIGITS    |
01AB|          | 00 |                               |
01AC| LD,C,A   | 4F | BC = ADDRESS TO BE EDITED     |
01AD| JP       | C3 | JUMP TO DISPLAY ROUTINE       |
01AE| NOP      | FC |                               |
01AF| NOP      | 01 |                               |
01B0| NOP      | 00 |                               |
01B1| NOP      | 00 |                               |
01B2| NOP      | 00 |                               |
01B3| NOP      | 00 |                               |
01B4| NOP      | 00 |                               |
01B5| NOP      | 00 |                               |
01B6| NOP      | 00 |                               |
01B7| NOP      | 00 |                               |
01B8| NOP      | 00 |                               |
01B9| LD,HL,nn | 21 | HL= 5TH TV OUTPUT LINE        |
01BA|          | 37 |                               |
01BB|          | F9 |                               |
01BC| LD,A,B   | 78 | A = HIGH 2 ADDRESS DIGITS     |
01BD| CALL     | CD | CALL 2 OUT                    |
01BE|          | 73 |   WRITE ADDRESS BEING EDITED  |
01BF|          | 00 |                               |
```

```
01C0| DEC HL    | 2B | ADJUST TV OUTPUT ADDRESS      |
01C1| DEC HL    | 2B |                               |
01C2| LD,A,C    | 79 | A = LOW 2 ADDRESS DIGITS      |
01C3| CALL      | CD | CALL 2 OUT                    |
01C4|           | 73 |   WRITE ADDRESS BEING EDITED  |
01C5|           | 00 |                               |
01C6| LD,HL,nn  | 21 | HL= ADDRESS OF "ENTER DATA"   |
01C7|           | 60 |                               |
01C8|           | 03 |                               |
01C9| CALL      | CD | CALL SCREEN WRITE             |
01CA|           | B8 |                               |
01CB|           | 02 |                               |
01CC| CALL      | CD | CALL GET ASCII                |
01CD|           | 34 |   GET COMMAND OR DATA         |
01CE|           | 00 |                               |
01CF| CP,A,n    | FE | CHECK IF INPUT = "X"          |
01D0|           | 18 |   (X IS FOR BACKUP)           |
01D1| JP,NZ     | C2 | JUMP IF NOT X                 |
01D2|           | D8 |                               |
01D3|           | 01 |                               |
01D4| DEC BC    | 0B | BACKUP ONE LOCATION           |
01D5| JP        | C3 | JUMP TO GET NEXT INPUT        |
01D6|           | FC |                               |
01D7|           | 01 |                               |
01D8| CP,A,n    | FE | CHECK IF INPUT = "S"          |
01D9|           | 13 |   (S IS FOR SKIP AHEAD)       |
01DA| JP,Z      | CA | JUMP IF S                     |
01DB|           | FB |                               |
01DC|           | 01 |                               |
01DD| CP,A,n    | FE | CHECK IF INPUT = "N"          |
01DE|           | 0E |   (N IS FOR NEW ADDRESS)      |
01DF| JP,Z      | CA | JUMP IF N                     |
01E0|           | 9F |                               |
01E1|           | 01 |                               |
01E2| CP,A,n    | FE | CHECK IF INPUT = "Q"          |
01E3|           | 11 |   (Q IS FOR QUIT)             |
01E4| JP,NZ     | C2 | JUMP IF NOT Q                 |
01E5|           | F7 |                               |
01E6|           | 01 |                               |
01E7| NOP       | 00 |                               |
01E8| CALL      | CD | CALL CLEAR                    |
01E9|           | 04 |                               |
01EA|           | 00 |                               |
01EB| JP        | C3 | JUMP TO START OF CMD LOOP     |
01EC|           | 14 |                               |
01ED|           | 01 |                               |
01EE| NOP       | 00 | RESERVED FOR EXPANSION        |
01EF| NOP       | 00 |                               |
01F0| NOP       | 00 |                               |
```

```
01F1| NOP        | 00 |                                  |
01F2| NOP        | 00 |                                  |
01F3| NOP        | 00 |                                  |
01F4| NOP        | 00 |                                  |
01F5| NOP        | 00 |                                  |
01F6| NOP        | 00 |                                  |
01F7| CALL       | CD | CALL GET 2 HEX AT START + 1      |
01F8|            | 13 |   TURN ASCII NOW IN A TO 1ST     |
01F9|            | 00 |   HEX (ITS DATA, NOT A CMD)      |
01FA| LD,(BC),A  | 02 | LOAD DATA TO MEMORY              |
01FB| INC BC     | 03 | MEM = NEXT LOC TO BE EDITED      |
01FC| DEC BC     | 0B |                                  |
01FD| DEC BC     | 0B | BC = BC - 4                      |
01FE| DEC BC     | 0B |                                  |
01FF| DEC BC     | 0B |                                  |
0200| CALL       | CD | CALL 40 OUT                      |
0201|            | 40 |                                  |
0202|            | 00 |                                  |
0203| INC BC     | 03 |                                  |
0204| INC BC     | 03 | BC = BC + 4                      |
0205| INC BC     | 03 |                                  |
0206| INC BC     | 03 |                                  |
0207| LD,A,(nn)  | 3A | DISPLAY LOCATION BEING           |
0208|            | 26 |   EDITED USING INVERSE           |
0209|            | F8 |     VIDEO                        |
020A| OR,A,n     | F6 | INVERT DIGIT                     |
020B|            | 40 |                                  |
020C| LD,(nn),A  | 32 |                                  |
020D|            | 26 |                                  |
020E|            | F8 |                                  |
020F| LD,A,(nn)  | 3A |                                  |
0210|            | 27 |                                  |
0211|            | F8 |                                  |
0212| OR,A,n     | F6 | INVERT DIGIT                     |
0213|            | 40 |                                  |
0214| LD,(nn),A  | 32 |                                  |
0215|            | 27 |                                  |
0216|            | F8 |                                  |
0217| JP         | C3 | JUMP TO DISPLAY ADDRESS          |
0218|            | B9 |                                  |
0219|            | 01 |                                  |
----+------------+----+----------------------------------+--------
021A| LD,HL,nn   | 21 | HL= ADDRESS OF "ENTER DATA"      |
021B|            | 60 |                                  |
021C|            | 03 |                                  | FIND
021D| CALL       | CD | CALL SCREEN WRITE                | ****
021E|            | B8 |                                  |
021F|            | 02 |                                  |
0220| CALL       | CD | CALL GET 2 HEX                   |
```

```
0221|           | 10 |   GET PATTERN TO SEARCH FOR |
0222|           | 00 |                             |
0223| LD,H,B    | 60 |                             |
0224| LD,L,C    | 69 |                             |
0225| LD,BC,nn  | 01 | BC = 00                     |
0226|           | 00 |                             |
0227|           | 00 |                             |
0228| CPIR      | ED | CHECK FOR A MATCH UNTIL     |
0229|           | B1 |   BC = 00                   |
022A| PUSH HL   | E5 | SAVE ADDRESS OF MATCH       |
022B| XOR,A,A   | AF | A = 0                       |
022C| CP,A,B    | B8 | CHECK IF NO MATCH (BC = 00) |
022D| JP,NZ,$+D | 20 | JUMP TO 023C IF NOT         |
022E|           | 0D |                             |
022F| CP,A,C    | B9 |                             |
0230| JP,NZ,$+A | 20 |                             |
0231|           | 0A |                             |
0232| POP BC    | C1 | BC= ADDRESS TO BE DISPLAYED |
0233| LD,HL,nn  | 21 | HL= ADDR OF "NO MATCH FOUND"|
0234|           | 70 |                             |
0235|           | 03 |                             |
0236| CALL      | CD | CALL SCREEN WRITE           |
0237|           | B8 |                             |
0238|           | 02 |                             |
0239| JP        | C3 | JUMP TO GET A NEW COMMAND   |
023A|           | 1E |                             |
023B|           | 01 |                             |
023C| POP BC    | C1 | BC = ADDRESS TO BE DISPLAYED|
023D| DEC BC    | 0B | ADJUST BC                   |
023E| JP,$+3D   | 18 | JUMP TO CALL 40 OUT         |
023F|           | 3D |                             |
----+-----------+----+-----------------------------+--------
0240| LD,HL,nn  | 21 | HL= ADDR OF "START ADDRESS?"|
0241|           | 80 |                             |
0242|           | 03 |                             |  MOVE
0243| CALL      | CD | CALL SCREEN WRITE           |  ****
0244|           | B8 |                             |
0245|           | 02 |                             |
0246| CALL      | CD | CALL GET 2 HEX              |
0247|           | 10 |   (GET HIGH HALF)           |
0248|           | 00 |                             |
0249| LD,B,A    | 47 |                             |
024A| CALL      | CD | CALL GET 2 HEX              |
024B|           | 10 |   (GET LOW HALF)            |
024C|           | 00 |                             |
024D| LD,C,A    | 4F | BC = STARTING ADDRESS       |
024E| LD,HL,nn  | 21 | HL= ADDR OF "END ADDRESS?"  |
024F|           | 90 |                             |
0250|           | 03 |                             |
```

```
0251| CALL       | CD | CALL SCREEN WRITE                |
0252|            | B8 |                                  |
0253|            | 02 |                                  |
0254| CALL       | CD | CALL GET 2 HEX                   |
0255|            | 10 |   (GET HIGH HALF)                |
0256|            | 00 |                                  |
0257| LD,D,A     | 57 |   (H USED IN GET 2 HEX)          |
0258| CALL       | CD | CALL GET 2 HEX                   |
0259|            | 10 |   (GET LOW HALF)                 |
025A|            | 00 |                                  |
025B| LD,L,A     | 6F | HL = ENDING ADDRESS              |
025C| LD,H,D     | 62 |                                  |
025D| PUSH HL    | E5 | SAVE HL                          |
025E| LD,HL,nn   | 21 | HL= ADDR OF "NEW START?"         |
025F|            | A0 |                                  |
0260|            | 03 |                                  |
0261| CALL       | CD | CALL SCREEN WRITE                |
0262|            | B8 |                                  |
0263|            | 02 |                                  |
0264| CALL       | CD | CALL GET 2 HEX                   |
0265|            | 10 |   (GET HIGH HALF)                |
0266|            | 00 |                                  |
0267| LD,D,A     | 57 |                                  |
0268| CALL       | CD | CALL GET 2 HEX                   |
0269|            | 10 |   (GET LOW HALF)                 |
026A|            | 00 |                                  |
026B| LD,E,A     | 5F | DE = NEW STARTING ADDRESS        |
026C| POP HL     | E1 | RESTORE HL                       |
026D| PUSH HL    | E5 |                                  |
026E| SBC,HL,BC  | ED | HL = HL - BC                     |
026F|            | 42 |                                  |
0270| LD,C,L     | 4D | COPY HL INTO BC (BC =            |
0271| LD,B,H     | 44 |   END ADDR - START ADDRESS)      |
0272| ADD,HL,DE  | 19 | HL = HL + DE                     |
0273| EX,HL,DE   | EB | EXCHANGE HL AND DE (NEW END)     |
0274| POP HL     | E1 | RESTORE HL (OLD END ADDRESS)     |
0275| INC BC     | 03 | ADJUST BC                        |
0276| LDDR       | ED | OLD ADDR > NEW ADDR UNTIL        |
0277|            | B8 |   BC (DIFFERENCE) = ZERO         |
0278| OUT,A,n    | D3 | WRITE A "BEEP"                   |
0279|            | 7F |   (ALL DONE)                     |
027A| INC DE     | 13 | ADJUST DE                        |
027B| LD,B,D     | 42 | COPY DE TO BC                    |
027C| LD,C,E     | 4B |                                  |
027D| JP         | C3 | JUMP TO GET COMMAND              |
027E|            | 14 |                                  |
027F|            | 01 |                                  |
----+------------+----+----------------------------------+--------
```

```
----+----------+----+-------------------------------+--------
0280| LD,SP,nn | 31 | TOP OF STACK = FFFF           |
0281|          | FF |                               |
0282|          | FF |                               |  POWER
0283| LD,A,n   | 3E | WRITE "AA" TO FIRST           |   UP
0284|          | AA |   STACK LOCATION              |  CHECK
0285| LD,(nn),A| 32 |                               |  *****
0286|          | FF |                               |
0287|          | FF |                               |
0288| LD,A,n   | 3E | WRITE "55" TO SECOND          |
0289|          | 55 |   STACK LOCATION              |
028A| LD,(nn),A| 32 |                               |
028B|          | FE |                               |
028C|          | FF |                               |
028D| LD,A,(nn)| 3A | READ FIRST STACK LOCATION     |
028E|          | FF |                               |
028F|          | FF |                               |
0290| CP,A,n   | FE | CHECK IF ITS AN "AA"          |
0291|          | AA |                               |
0292| JP,NZ    | C2 | JUMP IF NOT (MEMORY BAD)      |
0293|          | 9D |                               |
0294|          | 02 |                               |
0295| LD,A,(nn)| 3A | READ SECOND STACK LOCATION    |
0296|          | FE |                               |
0297|          | FF |                               |
0298| CP,A,n   | FE | CHECK IF ITS A "55"           |
0299|          | 55 |                               |
029A| JP,Z     | CA | JUMP IF IT IS (MEMORY IS OK)  |
029B|          | 00 |                               |
029C|          | 01 |                               |
029D| LD,HL,nn | 21 | HL= START OF SCREEN ADDRESS   |
029E|          | 00 |                               |
029F|          | F8 |                               |
02A0| LD,(HL),n| 36 | WRITE "S" TO SCREEN           |
02A1|          | 53 |                               |
02A2| INC HL   | 23 |                               |
02A3| LD,(HL),n| 36 | WRITE "T" TO SCREEN           |
02A4|          | 54 |                               |
02A5| INC HL   | 23 |                               |
02A6| LD,(HL),n| 36 | WRITE "A" TO SCREEN           |
02A7|          | 41 |                               |
02A8| INC HL   | 23 |                               |
02A9| LD,(HL),n| 36 | WRITE "C" TO SCREEN           |
02AA|          | 43 |                               |
02AB| INC HL   | 23 |                               |
02AC| LD,(HL),n| 36 | WRITE "K" TO SCREEN           |
02AD|          | 4B |                               |
02AE| INC HL   | 23 |                               |
02AF| LD,(HL),n| 36 | WRITE COLOR SPACE TO SCREEN   |
```

```
02B0|           | DF |                                  |
02B1| OUT,A,n   | D3 | WRITE A "BEEP"                   |
02B2|           | 7F |                                  |
02B3| CB,7,H    | CB | CHECK IF SCREEN IS FILLED        |
02B4|           | 7C | JUMP IF ITS NOT                  |
02B5| JP,NZ,$-7 | 20 |   (TO 02AF)                      |
02B6|           | F7 |                                  |
02B7| HALT      | 76 | HALT - MEMORY IS BAD             |
----+-----------+----+----------------------------------+--------
02B8| PUSH BC   | C5 | SAVE BC                          |
02B9| LD,DE,nn  | 11 | DE= SCREEN ADDRESS (LINE 1)      |
02BA|           | 57 |                                  | SCREEN
02BB|           | F8 |                                  | WRITE
02BC| LD,BC,nn  | 01 | BC= NUMBER CHARS. ON LINE 1      | ******
02BD|           | 08 |                                  |
02BE|           | 00 |                                  |
02BF| LDIR      | ED | WRITE 8 CHARS. TO LINE 1         |
02C0|           | B0 |                                  |
02C1| LD,E,n    | 1E | DE= SCREEN ADDRESS (LINE 2)      |
02C2|           | 77 |                                  |
02C3| LD,BC,nn  | 01 | BC= NUMBER CHARS ON LINE 2       |
02C4|           | 08 |                                  |
02C5|           | 00 |                                  |
02C6| LDIR      | ED | WRITE 8 CHARS. TO LINE 2         |
02C7|           | B0 |                                  |
02C8| POP BC    | C1 | RESTORE OLD BC                   |
02C9| RETURN    | C9 | RETURN                           |
02CA| NOP       | 00 |                                  |
02CB| NOP       | 00 |                                  |
02CC| NOP       | 00 |                                  |
02CD| NOP       | 00 |                                  |
02CE| NOP       | 00 |                                  |
02CF| NOP       | 00 |                                  |
----+-----------+----+----------------------------------+--------
02D0| UNUSED    | FF |                                  |
  .
  .
  .
02FF| UNUSED    | FF |                                  |
----+-----------+----+----------------------------------+--------
0300|     E     | 05 |                                  |
0301|     N     | 0E |  DATA FOR "SCREEN WRITE"         |
0302|     T     | 14 |                                  | DATA
0303|     E     | 05 |                                  | ****
0304|     R     | 12 |                                  |
0305|           | 20 |                                  |
0306|           | 20 |                                  |
0307|           | 20 |                                  |
0308|     A     | 01 |                                  |
```

0309	D	04
030A	D	04
030B	R	12
030C	E	05
030D	S	13
030E	S	13
030F		20
0310	E	05
0311	N	0E
0312	T	14
0313	E	05
0314	R	12
0315		20
0316		20
0317		20
0318	C	03
0319	O	0F
031A	M	0D
031B	M	0D
031C	A	01
031D	N	0E
031E	D	04
031F		20
0320	R	12
0321	U	15
0322	N	0E
0323	N	0E
0324	I	09
0325	N	0E
0326	G	07
0327		20
0328	P	10
0329	R	12
032A	O	0F
032B	G	07
032C	R	12
032D	A	01
032E	M	0D
032F		20
0330	S	13
0331	E	05
0332	L	0C
0333	F	06
0334		20
0335		20
0336		20
0337		20
0338	T	14
0339	E	05

033A	S	13
033B	T	14
033C	S	13
033D		20
033E		20
033F		20
0340	K	0B
0341	E	05
0342	Y	19
0343	B	02
0344	O	0F
0345	A	01
0346	R	12
0347	D	04
0348	T	14
0349	E	05
034A	S	13
034B	T	14
034C		20
034D		20
034E		20
034F		20
0350	R	12
0351	A	01
0352	M	0D
0353		20
0354		20
0355		20
0356		20
0357		20
0358	F	06
0359	A	01
035A	I	09
035B	L	0C
035C	U	15
035D	R	12
035E	E	05
035F	I	21
0360	E	05
0361	N	0E
0362	T	14
0363	E	05
0364	R	12
0365		20
0366		20
0367		20
0368	D	04
0369	A	01
036A	T	14

036B	A	01			
036C		20			
036D		20			
036E		20			
036F		20			
0370	N	0E			
0371	O	0F			
0372		20			
0373	M	0D			
0374	A	01			
0375	T	14			
0376	C	03			
0377	H	08			
0378	F	06			
0379	O	0F			
037A	U	15			
037B	N	0E			
037C	D	04			
037D		20			
037E		20			
037F		20			
0380	S	13			
0381	T	14			
0382	A	01			
0383	R	12			
0384	T	14			
0385		20			
0386		20			
0387		20			
0388	A	01			
0389	D	04			
038A	D	04			
038B	R	12			
038C	E	05			
038D	S	13			
038E	S	13			
038F	?	3F			
0390	E	05			
0391	N	0E			
0392	D	04			
0393		20			
0394		20			
0395		20			
0396		20			
0397		20			
0398	A	01			
0399	D	04			
039A	D	04			
039B	R	12			

039C	E	05	
039D	S	13	
039E	S	13	
039F	?	3F	
03A0	N	0E	
03A1	E	05	
03A2	W	17	
03A3		20	
03A4		20	
03A5		20	
03A6		20	
03A7		20	
03A8	S	13	
03A9	T	14	
03AA	A	01	
03AB	R	12	
03AC	T	14	
03AD	?	3F	
03AE		20	
03AF		20	

Bibliography

Bell Laboratories and Western Electric Engineering - "Semiconductor Memory" Allentown, Pennsylvania: AT&T Technologies Inc., 1983

Bursky, David - "S-100 Bus Handbook" Rochelle Park, New Jersey: Hayden, 1980

Hearst Business Communications, Inc. - "I. C. Master" Garden City, New York: Hearst Business Communications, Inc., 1983

Intel Engineering Staff - "Intel Component Data Catalog" Santa Clara, California: Intel, 1979

Texas Instruments Inc. Engineering Staff - "The TTL Data Book for Design Engineers - Second Edition" Dallas, Texas: Texas Instruments, Inc., 1976

Zilog Engineering Staff - "Z80 Assembly Language Programming Manual" Cupertino, California: Zilog, Inc., 1980

Zilog Engineering Staff - "Z80-CPU Technical Manual" Cupertino, California: Zilog, Inc., 1977

Douglas H. Rhyner was born in Pittsburgh, Pennsylvania on September fourth, 1958. He is the son of Glenn and Elizabeth Rhyner. Mr. Rhyner attended the University of Pittsburgh, from which he received a Bachelor of Science in Electrical Engineering on April 23, 1980. He is currently employed by AT&T Technologies, Inc. of Allentown, Pennsylvania, where he works as an Electrical Engineer dealing with Process Control computer systems. He is married to Lisa Patton Rhyner, and lives in Bethlehem, Pennsylvania.