PROTEC

THE PRO-80

ASSEMBLY

AND

OPERATIONS

MANUAL

NOTE;

PROTEC reserves the right to make changes at any time without notice to this publication and/or to the PRO-80 microcomputer. The information contained herein is believed to be accurate. However, PROTEC assures no responsibility for its user;nor any infrigements of patents or other rights of third parties which would result.

On deposit at the National Library 3rd trimister, 1981.

TABLE OF CONTENTS

INTRODUCTION

The Z-80 microprocessor has had a successful existance for 5 years. During this time the Z-80 has exceeded all hopes and expectations of the industry.

The powerful set of 158 basic instructions, (696 with the different addressing modes), the indexing capability and 16 bit arithmetic operation gives the Z-80 features that are found only in a minicomputer. The 8 addressing modes and 3 interupt modes combined with the block transfer instructions make the Z-80 a hard to beat 8-bit microprocessor. Furthermore, the 8080-A instruction set is a subset of Z-80 instructions. This permits programs written for 8080-A to be used directly with the Z80, allowing the user to choose from among the thousands of programs already available.

Manufacturers have lost no time in putting this powerful worker to the test. The microcomputers built around the Z-80 and oriented toward the small and medium sized business are multiplying with profusion, leaving a rather embarrassing amount of choice to eventual users of this type of computer. Meanwhile, there doesn't exist a truly economical and educational system that meets the needs of students, teachers, experimenters or anyone who wishes to know or evaluate at a reasonable price the performance of this wonderful machine, the Z-80.

It is precisely this void that we wish to fill in offering the PRO-80, we have designed it with care for maximum versatility, we have given it a S-100 bus allowing the user to expand his system at will by choosing from various modules already available on the market. We have provided wire wrap space for experimentation and building process control circuits on the same prime circuit board. The PRO-80 also has two parrallel input/output ports (Z80-PIO) permitting access to external peripheral equipment. These two ports possess 8 bits each, and each bit can be controlled by software. These assure the user control of 16 individual lines for particular applications. The Z80-PIO also has an internal interupt control and two pairs of lines for external exchanges (handshake). An interface for an audio cassette recorder provides the user with an economical means of recording programs and data directly on minicassette tapes.

The PRO-80 memory is made of 1 Kbyte of RAM expandable on the board to 2 Kbytes. A third Kbyte of EPROM contains the monitor which performs several powerful functions such as memory examine and change, register examine and change, next memory location,next alternate register and a single step operation mode, that provides you with the capability to execute and debug your program one instruction at a time. Other functions such as reset, program execute and cassette read-write are also featured on the PRO-80 monitor.

A hex keyboard, with 8 additional keys is used to load data and programs and to initiate the different functions of the monitor. Six "seven segment" digits are used to display the memory addresses, the Z-80 registers, the alternate registers and their contents.

The PRO-80 requires only a 5 volt, 1 ampere power supply. We have incorporated a 5V/1A voltage regulator so that only an 8 volt, 1 ampere transformer-rectifier is required to complete the PRO-80 power supply;this power pack is available through PROTEC.

Chapter 1 of this manual introduces the PRO-80 architecture. The basic components of the system are briefly described. Readers should however, have some background in microcomputers to fully understand the whole system design and operation. For beginners, this technical manual will be insufficient. Therefore, we refer them to a more detailed book, specially written for the Z-80, "Programming the Z-80" by Rodnay Zaks, Sybex.

Chapter II describes components such as resistors, capacitors and intergrated circuits. The beginner will learn to identify every component of the PRO-80. We have given all the instructions required for the assembly and testing of the PRO-80 microcomputer.

Chapter III describes the operation of the monitor. Every function of this program has been described, and several examples are given so as to better understand their mode of operation.

Chapter IV deals with three simulations: a light chaser, a traffic light and a digital clock.

I The PRO-80 Architecture

1.1 Introduction

The complete electronic diagram of the PRO-80 is shown in Annex 1. A simplified version of this diagram is given in figure 1.1 which explains the functions carried out by the main components of the system.

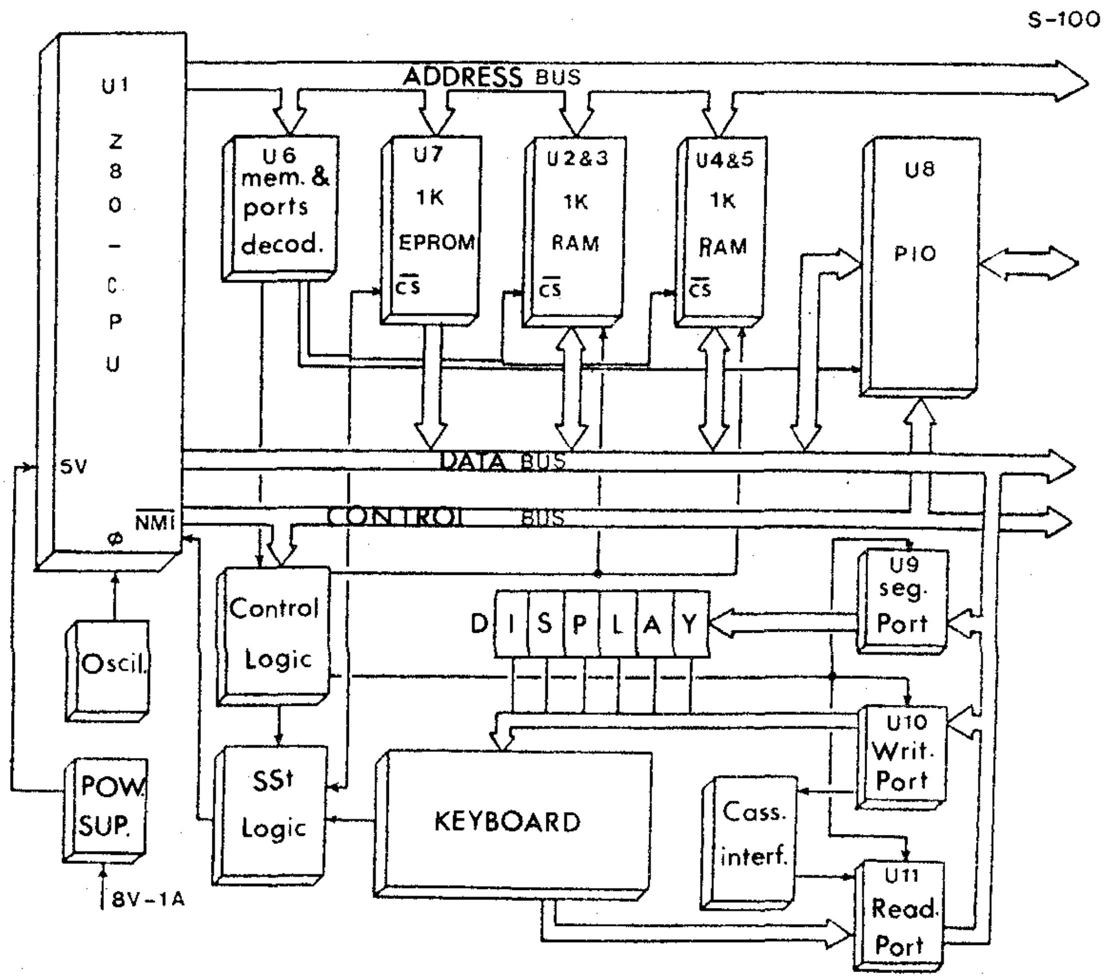


Figure 1.1: The PRO-80 Architecture

1.2 The Z-80

This microprocessor is made up of nearly 8000 transistors and is the brain of the PRO-80. It is almost a microcomputer in itself, consisting of three main units: the arithmetic and logic unit (ALU), the control unit and the internal memory unit.

Depending on the instruction being processed, the ALU can execute logic operations such as and, or, exclusive or, shift left, shift right and so on... It can also execute arithmetic operations such as increments, additions, etc...

The control unit is the system's management center, It controls the various steps of the process. Depending on the system's status and the instructions sequence, this unit generates the right signals to access external memories, to route all necessary information to the ALU, to the various registers, or to manage exchanges with the peripherals.

The Z-80 internal memory unit is the only one that features 22 registers for a total memory capacity of 207 bits. These registers are used to store information generated by the CPU or by any external component. This memory features (*):

- Two sets of identical 8-bit registers, each set consisting of an accumulator (A), a flag register (F) and three pairs of registers (B C, D E, H L) that can be used separatily or in pairs as 16-bit registers.

- An 8-bit interrupt register (I) which allows a minimal access time to a service routine in any memory location. In the interrupt mode, the I-register holds the high byte of the routine address. The low byte is generated by the peripheral requesting the interrupt.

- 7-bit register (R) is used to refresh the dynamic memories.

- Two 16-bit registers (IX and IY) are used for indexed addressing. In this addressing mode, the register (either IX or IY) contains a reference memory location. An additional byte included in the indexed instructions gives the offset relative to the reference address

- A stack pointer (SP), which holds a 16-bit address of the top stack located in an external system RAM memory. The stack operates in a "last in, first out" manner which allows the last information added to the stack (PUSHED) to be the first removed (POPED).

(*) : See cover of the "Microreference Manual", Mostek.

- At last, the program counter (PC) which holds the 16-bit address of the current instruction being fetched from the memory.

Further information can be found in the "Z-80 CPU Technical Manual" by Mostek or Zilog.

1.3. MEMORIES

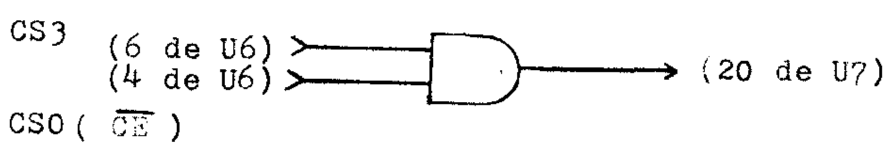
Memories are made up of individual cells, capable of storing information. The memory capacity is expressed in binary digits called bits, or more often in bytes consisting of 8 bits. A 1 Kbyte memory consists of 2^{10}(i.e. 1024) bytes of 8 bits each.

There are two basic categories of memory. The first one is the read only memories (ROM) which stores information permanently: its content remaining always unaltered even when the electric power supply is shut off. The ROM content is wired in by the manufacturer. However, different kinds of ROM such as PROM and EPROM allow the user to pattern his own ROM. The PROM can be programmed by the user while the EPROM are erasable and reprogrammable, provided a special procedure is followed.

The second category of memory is the RAM (Random Access Memory) in which reading and writing operations are software controllable and information is randomly accessed. In this type of memory, information is lost when the electric power is shut off.

The PRO-80 has 1 Kbyte EPROM (U7) that holds the monitor program even when the power is shut off. A user 1 Kbyte RAM (U2 and U3) has been provided and allowance has been made for an extra Kbyte of RAM (U4 and U5).

The more experienced reader can easily obtain an extra Kbyte of EPROM for his own personal use: Instead of grounding pin #19 of the 2716 chip (2 Kbyte EPROM), this pin should be connected to A-10. The following change would then be necessary to allow access to the second Kbyte of EPROM:



$\frac{1}{4}$ 74LS08 (not supplied)

1.3.1 MEMORY MAPPING

Table 1.1 is a summary of a memory map. We can see with it that the monitor is located in the first Kbyte of memory (address locations 0000H(1)to03FF H. It also uses the last 122 bytes of the RAM (locations 1386H to 13FFH). Therefore, the programmer is not allowed access to these locations. The 0400H to OFFF addresses are not used by the system except when a second Kbyte of EPROM is made available. The user's RAM is located between 1000H and 1385H. Locations 1386 to 13BA are used for both user and monitor stack pointers. Locations 13BBH to 13CFH are used to route the RST instructions (see annex3). The optional Kbyte of RAM is located between 1400H and 17FFH.

Locations	Functions	Comments
0000H to 03FFH	Monitor (U7)	Executes the PRO-80 functions
0400H to 0FFFH	Unused	Except when the second Kbyte of EPROM is made available(0400 to 07FFH).
1000H to 1385H	RAM(U2-U3)	User's memory area
1386H to 13A0H	User's stack pointer	RAM locations for user's stack
13A1H to 13BAH	Monitor's stack pointer	RAM locations for monitor's stack
13BBH to 13CFH	Monitor variables	For more detail, refer to Annex 2
1400H to 17FFH	RAM(U4-U5)	1 optional Kbyte of RAM
1800H ...	Unused	

Table 1.1: Memory Map List

(1):H: Hexadecimal.

This memory mapping requires an address decoder. This is carried out by the 74LS139 chip (U6) which also decodes the PRO-80 input/output ports.

1.4. PORT DECODING

In microcomputers, the information exchange between the CPU and its peripherals is handled by the input/output ports. Port addressing in the Z-80 requires 8 bits. This gives a possibility of 2^8 or 256 addressable ports. The PRO-80 uses 4 blocks consisting of 4 ports each detailed as below:

Addresses	I/O Ports
40H to 43H	PIO
44H to 47H	Hexadecimal keyboard
48H to 4BH	Display digits
4CH to 4FH	Display segments

Table 1.2: Input/output ports organization.

Keyboard reading, digit display and segment setting functions actually require only one port each. However, to simplify our design, we have dedicated 4 ports to each function, ports 44H to 47H therefore have exactly the same function, and either one can be used to initiate unit 11. The same applies to ports 48H to 4BH (U10) and to ports 4CH to 4FH (U9).

The PIO (Parallel Input/Output) however requires 4 separate ports. Their locations are given in the following paragraph.

1.5. The Z80-PIO

The PIO has two input/output parallel ports (A&B), these ports are TTL compatible and can interface the PRO-80 with an ASCII keyboard, video monitor, card reader, printer and with a wide range of other peripherals. Each port has a control register that allows it to operate as an 8-bit input port or an 8-bit output port. Port A can also operate as an 8-bit bidirectional bus. Programming of such control registers is summarized on page 25 of the "Micro-Reference Manuel". Further information can be found in the"Z-80-PIO Technical Manual" by Mostek/Zilog .

The PRO-80 input/output ports and control registers are decoded as shown below.

ADDRESS LOCATION	PORTS
40H	A Data Register
41H	B Data register
42H	A Control Register
43H	B Control Register

Figure 1.3: PIC port addressing

For more convenience, the data lines and "handshake" control signals have been connected to the S-100 bus on the PRO-80.

1.6. KEYBOARD SCANNING AND HEX DISPLAY

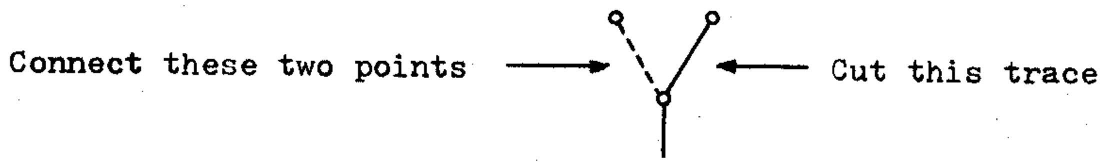
The PRO-80 monitor executes these 2 functions simultaneously. When the "RES" key is depressed, the monitor resets the RAM variables and starts the hex keyboard and display scanning.

Data is sent to the display via unit 9 (port 4CH). The common cathode of the digit to be lighted is reset via unit 10 (port 48H). The logical "0" received from this port is also applied to a keyboard row. The digit is held on for a while and the monitor takes a reading of port 44H (unit 11) to check if a key in that row is depressed. New data is sent to the next digit, the monitor takes another reading of port 44H and the keyboard/display scanning continues.

1.7. CASSETTE TAPE INTERFACE

The audio frequencies recorded by a cassette tape are much lower than those used in microcomputers. To be recorded the information must be made audible before it can be applied to the microphone or auxiliary input of a cassette recorder. The PRO-80 uses a format similar to the Kansas City Standard for program recording: "1"s are coded by a 2400Hz signal,"0"s by a 1200Hz signal and the data rate is approximately 300 bauds. Each recorded byte contains one start bit and one stop bit (both are zeroes). A recording of null bytes serves as a program delimiter(a 10 second pause as leader and a 5 second pause as trailer). Programs are recorded serially via data line D6 of port 48H (pin #20 of UIO). The signal is applied to low pass filter for high frequency suppressing. A voltage divider allows the cassette recordingthrough the mic or auxiliary input. The PRO-80 comes wired with an output which must be connected to the auxiliary input of the cassette recorder.

In order to use a mic input, the following changes must be done to the traces located under R41:



The cassette RAM transfer is done through the recorder's earphone output. The signal is squared by an amplifier inverter and then decoded by U20 and U19.

A second signal applied to pin #14 of U11 is used by the monitor for synchronization purposes during its serial to parrallel conversion.

1.8 SINGLE STEP

This fonction is carried out by a special logic under the monitor control. It provides a powerful debugging tool, allowing the programmer to execute the program under development one instruction at a time. Depressing the "SST" key causes output "3" of U22 to go low. The falling edge of this signal is used to transfer a "0" to the output Q of U21 (input 9 of U23). As soon as the execution of an instruction contained in the RAM begins, "$\overline{CE}$" of the PROM becomes high. It is inverted by U24, and a "0" is applied to the second input of U23 (pin #10). The output of this gate then becomes low. At the falling edge of $\overline{M1}$ generated by the Z-80 during the fetch cycle, the "0" is transfered to the input A of the U20 flip-flop. This FF generates a pulse on the non-maskable interrupt of the Z-80 ($\overline{NMI}$). As soon as the instruction has been executed, control returns to the monitor, the address of the next instruction and the accumulator content are displayed.

1.9 OSCILLATOR

The clock signal that runs the Z-80 is generated by an oscillator using two inverters of U24. This oscillator is controlled by a 4MHz crystal. The output signal is divided by 2(U19) and applied to the Z-80 input clock.

1.10 POWER SUPPLY

The 7805 (U18) built into the PRO-80, generates a regulated 5 Volts at 1 Amp. It needs only 7.5 to 8.5 Volts, 1 Amp. transformer rectifier and a 2200 µF/16V electrolytic capacitor, to complete the Pro-80's power supply. Capacitors C4, C5 & C6 are used to maintain a more stable output voltage.

1.11 S-100 BUS

Standardization has been sacrificed to increase flexibility so that almost all control signals used by the PRO-80 have been connected to the S-100 bus and are properly identified by a legend. The user can cut traces of unnecessary signals. He also can modify these signals in the wire wrap area, create other signals, connect them to the S-100 bus, etc... The Pro-80 offers maximum flexibility. The only limit is your imagination.

II ASSEMBLY

2.1. INTRODUCTION

Readers with some prior experience in the field of electronics may want to glance at the contents of the next four paragraphs. Others, however, should pay paticular attention to every detail given below. They can start to assemble the circuit only when each component has been identified.

2.2. RESISTANCES

Resistances are defined by their value in ohms (Ω), Or in kiloohms ($1K\Omega = 1000\Omega$) and by the tolerance relative to this value. Each resistance has different color bands to identify its value and tolerance (See figure2.1)

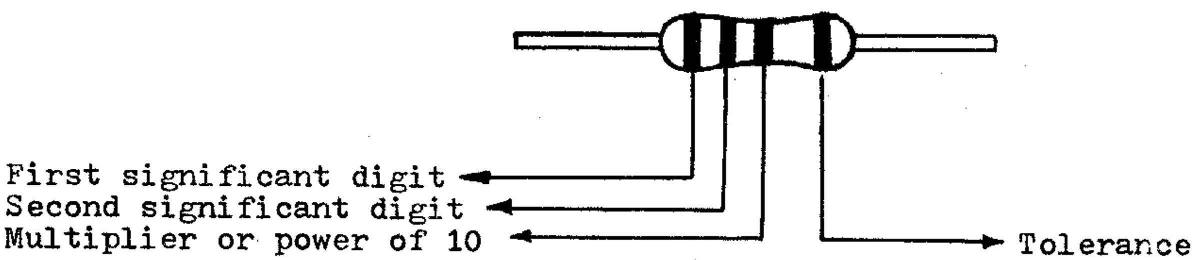


Figure 2.1. Resistance value and tolerance code.

The first and second bands at the end of a resistance represent the first and second significant digits respectively, the third band indicates the multiplier factor(or power of 10); the fourth can be gold or silver and identifies the tolerance given to the resistance value. The following table lists the color code of resistances :

COLOR	SIGNIFICANT DIGIT	MULTIPLIER	TOLERANCE
black	0	1	—
brown	1	10	—
red	2	100	—
orange	3	1,000	—
yellow	4	10,000	—
green	5	100,000	—
blue	6	1,000,000	—
purple	7	10,000,000	—
gray	8	100,000,000	—
white	9	1,000,000,000	—
gold	—	—	5%
silver	—	—	10%

Example: The value of a resistance with yellow, violet, red and gold bands is : $47 \times 100\Omega \pm 5\% = 47 \times 10^2\Omega \pm 5\%$ or

$$4.7\ k\Omega \pm 5\%$$

2.3 CAPACITORS

There are different types of capacitors. The PRO-80 kit includes a tantalum capacitor and several disc capacitors. (See figure2.2 for proper identification.)

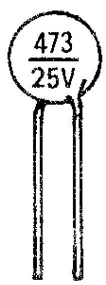
Figure 2.2.: Capacitors used in the PRO-80 kit.

The value of disc capacitors is generally given with its maximum allowable voltage. This value can be expressed directly in micro Farads (μF)

0.068 μF capacitor, maximum 30 volts

It can also be expressed in pico Farads (pF), with two significant digits followed by a multiplier

Example



Capacity value of 47×10^3 pF= 47000pF =.047 μF, 25 volts maximum.

Note: 1μF = 1,000,000 pF

2.4. INTEGRATED CIRCUITS

Integrated circuits do not all look alike. The most common type is called a DIP or "Dual In Line Package". DIP IC pins are numbered, so we must first identify pin #1 and by travelling counterclockwise determine pin numbers 2,3,etc... Several standards are used to identify pin #1. The following figure indicates most of these standards.

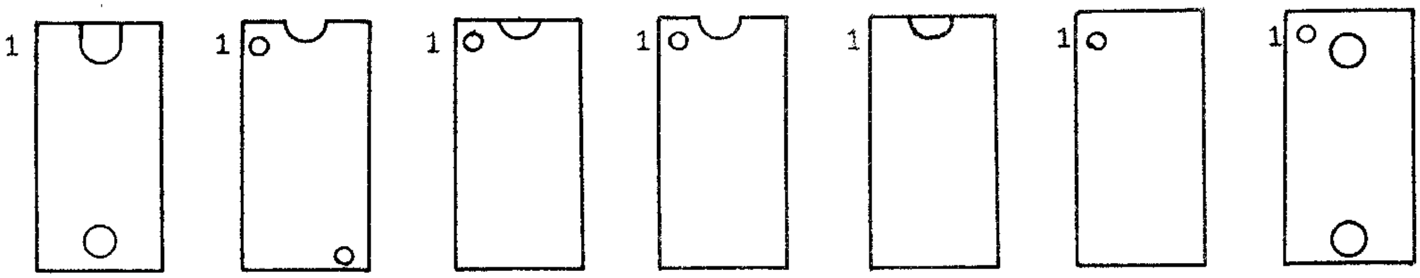


Figure 2.3.: Orientation of intergrated circuits

Note that it is essential to determine the correct position of integrated circuits before inserting them in their sockets. In the Pro-80, all IC's are placed so that pin #1 (identified by a white dot on the printed board) is at the upper left hand corner when the hex keyboard is at the lower right hand corner...

2.4.1 INTEGRATED CIRCUIT MOUNTING

Integrated circuits usually come packaged with their rows of leads spread apart so that the space between the leads is slightly larger than the space between the socket holes in which the leads are to be inserted. Straighten the leads at 90° before inserting them into their socket holes. This must be done on a flat surface covered with a piece of aluminum foil (preferably grounded). IC pins such as those of the 2716, 2114 , Z-80 CPU or Z-80-PIO should never be touched, otherwise these IC's would be damaged.

2.5. SOLDERING

Up to 90% of the defects are due to a bad soldering joint. The job must be done perfectly. We suggest that beginners practice with pieces of wire before trying their skills on the PRO-80 components.

Use a lead and tin solder (60/40 ratio), 20 or 22 guage. NEVER use an acid solder, it will corrode and rapidly damage printed circuits rendering the system inoperative.

The soldering iron must not be a heavy duty type: a 30 to 40 watt iron is enough for electronic soldering. The iron must be kept clean by wiping its tip with a damp sponge or cloth.

Components must be rid of all foreign matter before soldering. Place the tip of the iron on the spot where the wire and printed circuit meet. The tip must rest firmly on both elements. Apply the solder and watch carefully: the iron must be removed as soon as the compound starts to spread around the connection. Too much soldering can bridge two traces of printed circuits. Remove the iron and check the connection. It must be shaped like a funnel, be smooth and shiny. A "cold solder" looks rather dull and is often due to insufficient heat.

On the other hand, too much heat may damage the components and the board. Try to find a happy medium. The iron should be applied no more than two or three seconds. After each soldering, cut the excess wire as close as possible to the connection and start over for the next connection.

2.6. ASSEMBLY

Consult your layout and parts list to identify each PRO-80 component . Get your iron and follow these directions:

1. Each north-south end of the printed board has three holes in which to insert the feet of the PRO-80; tighten the screws finger tight.

2. Mount and solder the following resistances:

a- 10K, $\frac{1}{4}$W, 5% (Brown-Black-Orange)
.R14, R15, R16, R13, R12, R11, R10, R9, R7, R8, R6, R1, R2, R3, R4, R5, R18, R19, R22, R20, R17, R21

b- R28, R29, R30, R31, R25, R26, R39

c- 10, $\frac{1}{4}$W, 5% (Brown-Black-Black)
.R32, R33, R34, R38, R36, R35, R37.

d- 1K, $\frac{1}{4}$W, 5% (Brown-Black-Red)
.R23, R24, R40

e- 2K, $\frac{1}{4}$W, 5% (Red-Black-Red)
.R27. R45

f- 24K, $\frac{1}{4}$W, 5% (Red-Yellow-Orange)
.R46

g- 27K, $\frac{1}{4}$W, 5% (Red-Purple-Orange)
.R44

h- 100K, $\frac{1}{4}$W, 5% (Brown-Black-Yellow)
.R41

i- 4.7K, $\frac{1}{4}$W, 5% (Yellow-Purple-Red)
.R42

j- 120, $\frac{1}{4}$W, 5% (Brown-Red-Brown)
.R43

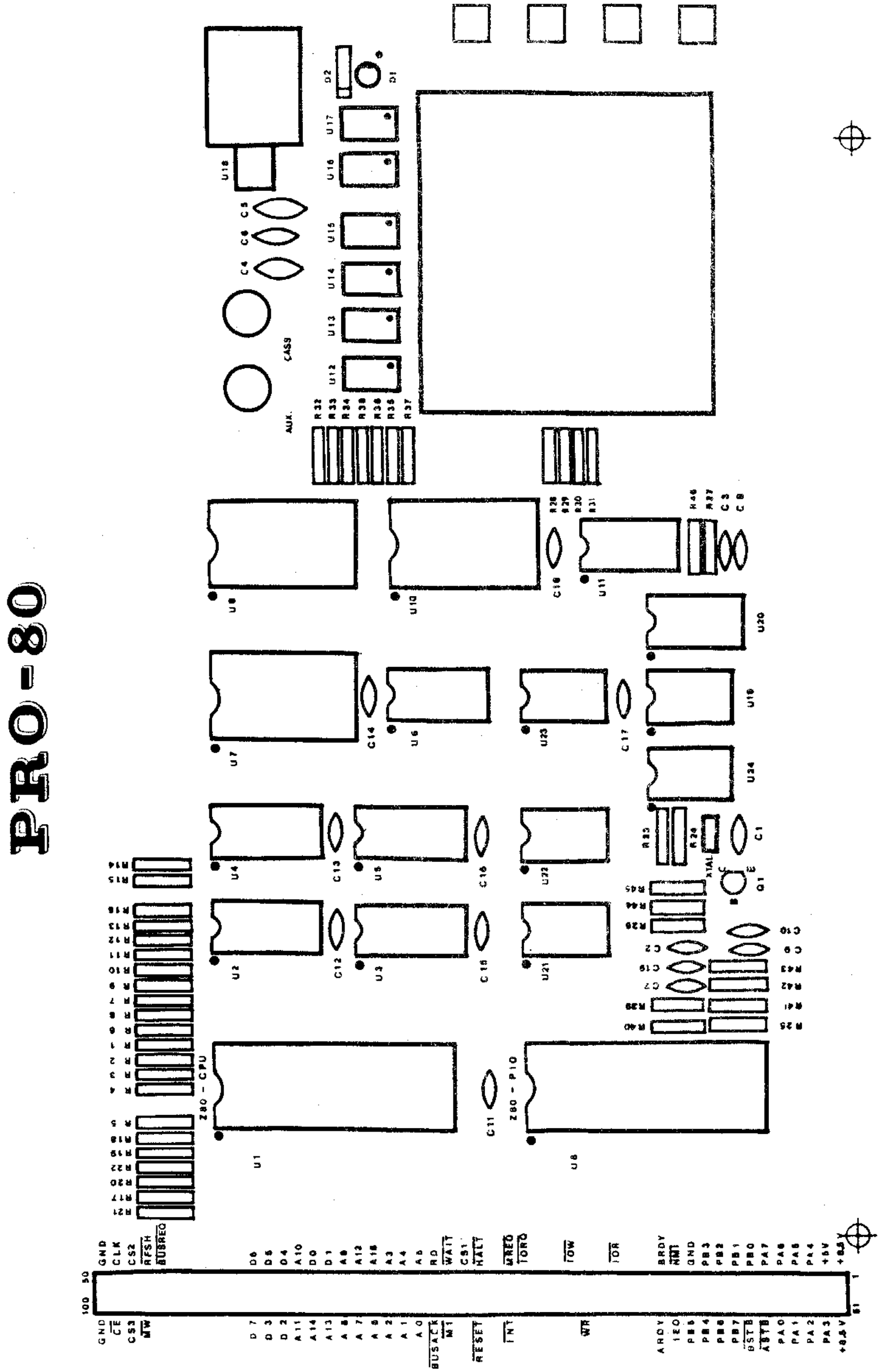


Figure 2.5 : PRO-80 LAYOUT

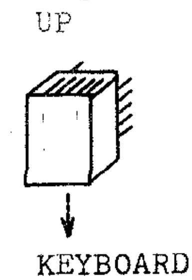
PARTS LIST

LABEL	QUANTITY	DESIGNATION	COMMENTS
U1	1	Z-80 CPU	Microprocessor
U2, U3	2	2114	1Kx4bit RAM
U6	1	74LS139	Dual decoder 2 to 4
U7	1	2716 (or 2516)	EPROM, 2Kx8bits, 5V
U8	1	Z-80 PIO	Parallel I/O ports
U9, U10	2	74S412	8bit buffer register
U11	1	74LS367	8bit latch register
U12-U17	6	FND367	7 segment digits
U18	1	7805	5volt regulator
U19, U21	2	74LS74	D flip-flops
U20	1	74LS221	Dual latch
U22, U23	2	74LS32	Quad OR gates
U24	1	74LS04	Hex inverters
Q1	1	2N3904	NPN transistors
R1-22; 25, 26, 28-31; 39	29	Resistances 10K.	¼ watt, ± 5%
R23, 24, 40,	3	Resistances1K	¼ watt, ± 5%
R27, R45	2	Resistances2K	"
R32-R38	7	" 10	"
R41	1	" 100K	"
R42	1	" 4.7K	"
R43	1	" 120	"
R44	1	" 27K	"
R46	1	" 24K	"
C1	1	Capacitor 22opF	Ceramic
C2	1	" 0.47uF	Ceramic
C3	1	" 0.001uF	Ceramic
C5	1	" 22 uF	Tantalum
C7, C10	2	" 0.005uF	Ceramic (or 0.0047)
C8	1	" 0.02uF	Ceramic
C9	1	" 0.047uF	Ceramic (Or 0.05)
C4, C6, C11-C19	11	" 0.1uF	Ceramic
XTAL	1	Crystal	4Mhz
D1	1	TIL 220	Light emitting diode
D2	1	1N914	Switching diode
	2	40 pins socket	
	3	24 pins socket	
	4	18 pins socket	
	3	16 pins socket	
	5	14 pins socket	
	1	keyboard with legend	
	4	Push buttons	
	2	Phono jacks	
	6	nuts & screws	

3. Mount and solder the following IC chip sockets :

a- 14 pins : U21-U22-U23-U24-U19
b- 16 pins : U6-U11-U20
c- 18 pins : U2-U3-U4-U5
d- 24 pins : U7-U9-U10
e- 40 pins : U1-U8

4. Mount and solder the 7 segment digitsas follows



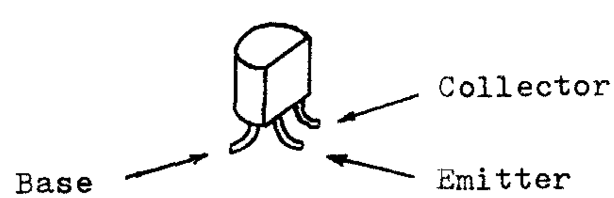
- U12 - U13 - U14 - U15 - U16 - U17

5. Mount and solder the D2 diode (band on left hand side) and the D1 LED the groove must be oriented towards the white dot on the printed circuit

6. Mount and solder the following capacitors:

a- 0.1µF	:	C4, C6, C11, C12, C13, C14, C15, C16, C17, C18, C19
b- 220pF	:	C1
c- 0.47µF	:	C2
d- 0.001µF	:	C3
e- 22 µF	:	C5, the red dot must be placed <u>downwards</u> .
f- 0.005µF	:	C7, C10
g- 0.02µF	:	C8
h- 0.047µF	:	C9

7. Mount and solder the XTAL crystal and Q1 transistor. The transistor should be placed as indicated below :



8. Mount and solder the U18 regulator.

9. Mount and solder the four push button keys.

10. Before mounting the keyboard, carefully straighten its pins placing them so that they appear on the flip side of the board, then apply enough solder to secure the connection.

Identify each key as in figure 2.6.

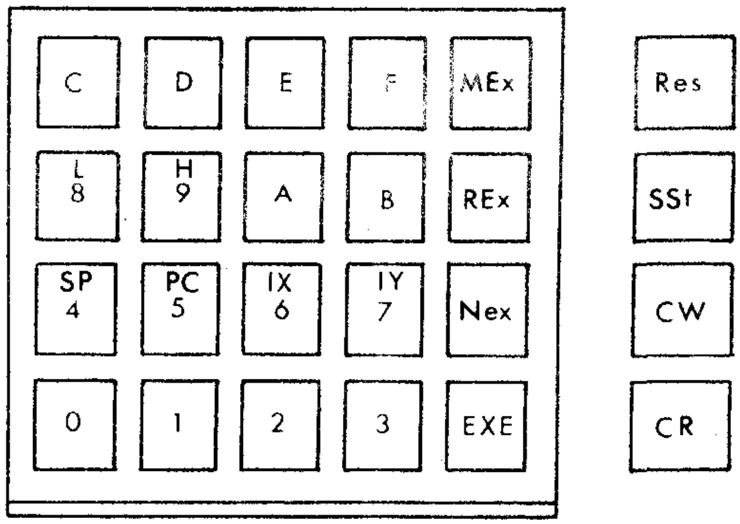


Figure 2.6 : Keyboard Layout

11. Mount and solder the audio connectors as illustrated in the figure below.

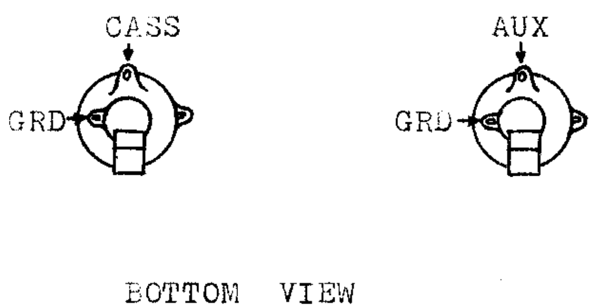


Figure 2.7 : Audio Connectors Mounting.

12. If the second Kbyte of EPROM is not used, ground pin #19 of U7. (Solder bridge pins #19 & 20 of U7)

2.7. PRELIMINARY CHECKING

Connect an 8 volt, 1 ampere D.C. supply to the two feeding points in the upper right hand corner of the PRO-80. Please check the polarity (the + is up and the - is down). If you use the PROTEC power supply pack, the white wire must be connected to the +8 volts. When the power supply is connected, the third pin of U18 (display digit side) must be at +5V relative to its middle pin (ground). If that is not the case, check the solder joints and make sure there is no short circuits. DO NOT proceed to the next step before obtaining the required voltage.

Shut off the power supply and mount the integrated circuits observing the proper orientation (pin #1 on the same side as the white dot on the board).

- U2-U3.. 2114
- U6...74LS139
- U19-U21..74LS74
- U22-U23..74LS32
- U24..74LS04
- U20..74LS221
- U11..74LS367
- U10-U9...74S412
- U7...2716 (or 2516)
- U1...Z-80 CPU
- U8...Z-80 PIO

Double check the orientation of the integrated circuits and make sure all pins fit securely in their holes.

The PRO-80 is now ready. Turn it on and depress the "Res" Push button. The " ⌐ " prompt symbol must appear in the first leftmost digit position.

Everything is in order. Perfect ! Proceed to the next chapter.

If the " ⌐ " prompt symbol does not appear, shut off the power and double check the connections. Unwanted bridges, cold solder joints and overlooked connections are often to blame.

If you can't find any fault in the connections and if you don't have the proper troubleshooting equipment, contact your dealer or write to PROTEC at:

B.O. 271
St-Laurent Branch,
Montreal (Quebec)
Canada.
H4L 4V6

III THE MONITOR

3.1. INTRODUCTION

If the microprocessor is the brain of any microcomputer, the monitor program is its heart . Without a heart, the brain cannot function on its own. The monitor complexity may vary, but this program is always there to decode and manage the input/output transfer, and to help the programmer to load and debug his program. The PRO-80 monitor is stored in a non-volatile 1 kilobyte memory and allows the programmer to have a complete control over every function as further discussed below.

3.2. THE RES PUSH BUTTON KEY

When this key is depressed, the processor is reset and the program execution begins at the first starting address 0000H. The PRO-80 monitor is located in the first kilobyte of the system's memory, so the 0000H address contains its first executable instruction. Depressing the RES key will cause the execution of the monitor program. The monitor initiates the user's stack pointer to address location 13A0H, resets the RAM variables and enters the MEMORY EXAMINE MODE. The system displays the " ⌐ " prompt symbol and starts scanning the keyboard for any new user entry.

3.3 MEMORY EXAMINE

This function is used to enter a new program or to examine and change the content of a memory location. The memory examine mode is used as below:

- Depress the MEX key. The monitor responds by displaying the " ⌐ " prompt symbol.
- Key in the four hex digit memory address (high digit first).
- Depress the NEX key. The content of that memory location is displayed in the two data digit positions.

If the NEX key is depressed before the four address digits have been entered, no data is displayed and the monitor waits for the remaining address digits.

Displayed data can be changed simply by keying in new data and depressing the NEX key. This key causes the monitor to update the content of the memory location. The address is incremented by one and the content of this new location is displayed.

<u>IMPORTANT</u>:

When the PRO-80 has a one kilobyte RAM, the user's memory is located between 1000H and 1385H. (Consult the memory map list)

3.3.1. APPLICATION

Connect the PRO-80 to an +8 Volt, 1 ampere power supply and depress the following keys:

Key	Address display	Data display	Comment
- RES	⊓ _ _ _	_ _	
-110	1 1 0 _	_ _	
-NEX	1 1 0 _	_ _	Incomplete address: nothing happens. the monitor waits for the fourth digit.
- 0	1 1 0 0	_ _	
- NEX	1 1 0 0	X X	The two digits XX represent the content of memory address location 1100H. Change by keying in a new data. EX: 12H
- 1	1 1 0 0	1 X	
- 2	1 1 0 0	1 2	12H is displayed but the location 1100H is updated by the monitor only when the NEX key is depressed. In case of error, new data canbe keyed in as many times as necessary. Depressing the NEX key validates the data.
- NEX	1 1 0 1	X X	12H data is transferred to 1100H location. The memory address is incremented and the content of new address is displayed..
- MEX	⊓ _ _ _	_ _	Return to memory examine mode.
- 1100	1 1 0 0	_ _	

- NEX	1 1 0 0 | 1 2	The new content of the memory location is displayed.

The procedure is simple and will be easily remembered if you try it out two or three times with your own data.

3.4. REGISTER EXAMINE

The PRO-80 uses an image register for each Z-80 register accessed by this function. The monitor transfers the content of the image registers to the microprocessor corresponding registers before each program execution, or single step execution. Conversely, the content of the Z-80 registers is retransmitted by the monitor to the image registers after each instruction is executed in the SINGLE STEP mode. This enables the user to :

1. Load a register before a program execute.
2. Examine or change during an execution in the single step mode, the content of the 8-bit registers A,B,C,D,E,F, and their corresponding alternate registers; the content of the 16-bit registers IX and IY, as well as those the program counter PC and the stack pointer SP. The procedure is described in the following paragraphs:

3.4.1. 8-BIT REGISTERS

- Depressing REX key initiates the register examine mode. The letter " r " appears in the high address digit display.
- Key in the desired register.
- Depress the NEX key. The content of the register is displayed in the two data digit positions.
- To modify the content, key in the new data and depress the NEX key.
- The monitor updates the content, displays the alternate registers and its content with the letter A (alternate) in the Low address position.
- The same basic steps apply to change the alternate register content.
- The NEX key initiates a new cycle which begins with the starting register.
- Repeat by depressing the REX key to examine or change the content of other registers or alternate registers.

3.4.2 APPLICATION

The purpose of this exercise is to examine and change the contents of registers A and alternate A, and to check their new contents which will be 1AH and 3EH respectively.

- REX	r						Register examine mode
- A	A						Selected register
- NEX	A				X	X	(A)$^{(1)}$= XX
- 1A	A				1	A	The new content of A is displayed but is not yet recorded in the image register.
- NEX	A			A	X	X	The content of A is now 1AH; the alternate A register is automatically displayed with its content.
- 3E	A			A	3	E	The content of A' is not yet updated: new data can be keyed in to correct any error.
- NEX	A				1	A	Uptade of (A'),(A)=1A
- NEX	A			A	3	E	(A')=3E
- REX	r						Return to the REX to examine or change the content of another register.

NOTES:

1. The prime mark "'" indicates the alternate register.
2. It is important to remember that the monitor makes changes on the image registers, and a program execute or single step execute is required for effective changes in the microprocessor registers.

(1) the brackets denote the <u>content</u> of register(or memory location).

3.4.3 16-BIT REGISTERS

The same basic steps apply. The only difference is that the content of these registers is expressed by 4 hexadecimal digits. Use the following procedures:

- Depress the REX key.
- Key in the desired register.
- Depress the NEX key. The monitor displays the high byte of the register's content. This byte can be changed using the procedure described above for the 8-bit register.
- Depressing the NEX key again causes the monitor to update the high byte and to display the low byte with the "__" identification symbol.
- Data is modified following the same procedure.
- Depressing the NEX key a third time causes the low byte to be updated and the high byte to be displayed. The cycle is repeated.
- Depress the REX key and repeat the same procedure to examine or change the content of other registers.

3.4.4. APPLICATION

The purpose of the exercise is to examine and change the content of the program counter (PC) and to check its new content. We will be changing the content to 1000H.

- REX	r						Examine register mode
- PC (5)	5						Selected register
- NEX	5				X	X	The high byte in the PC is XX.
- 10	5				1	0	The new byte is displayed but is still not recorded.
- NEX	5			-	X	X	The high byte has been updated and the low byte is displayed
- 00	5			-	0	0	The new value of the low byte is now displayed.

- NEX	5				1	0	The low byte is updated and the high byte is displayed with its new content.
- NEX	5			-	0	0	The new value of the low byte is 00H.
- EXR	r						Return to the REX mode to examine and change another register.

3.5 NEXT MODE

This function is used in the REX mode and the MEX mode. The basic steps have already been explained in paragraphs 3 and 4 and are summarized below.

3.5.1. MEX MODE

. Depressing the NEX key causes the content of a memory address to be displayed

. When the NEX key is depressed a second time, the monitor executes the following operations :

- It updates the content of the addressed memory location.

- It increments the address.

- It displays the incremented address with its content

3.5.2 REX MODE

3.5.2.1. 8-BIT REGISTERS :

. Depressing the NEX key causes the content of the register to be displayed.

. When the NEX key is depressed a second time, the content displayed in the two data digits is recorded in the image register. The alternate register is then displayed with its content and the letter A (Alternate).

. When the NEX key is depressed a third time, the monitor updates the alternate image register and displays the content of the starting register. The cycle is repeated.

3.5.2.2. 16-BIT REGISTERS

- Depressing the Nex key causes the high byte of the register content to be displayed.
- When the key is depressed a second time, the high byte is updated and the low byte is displayed with the "__" symbol.
- When the key is depressed a third time, the low byte is updated and the new value of the high byte is displayed as in the beginning of a new cycle.
- Practice using your own data to get the feeling of the NEX mode.

3.6. EXECUTION

All programs are merely a set of instructions to be executed sequentially. Sequencing is done by the program counter (PC) which contains at any given time the address of the next program instruction to be executed. Before the EXe key is activated, the PC must be loaded with the address of first program instruction. Once the EXe key is depressed, control remains in the user's program right to the end of the execute operation or until the RES key is depressed.

3.6.1. APPLICATION

The purpose of this exercise is to load 40H into A register, to add OEH to the A content and to store the result into the 1100H memory location. These operations are to be executed by the following program

ADDRESS	MACHINE CODE	MNEMONIC	COMMENTS
1000	3E	LDA, 40H	Load 40H into the
1001	40		A-register
1002	C6	ADDA, OEH	Add OEH to the con-
1003	OE		tent of A and record the result in A.
1004	32	LD(1100),A	Transfer the A con-
1005	00		tent to the 1100H
1006	11		memory location.
1007	76	HALT	STOP

The MEX and NEX modes are used to load this program as described in paragraph 3.3.

Key	Address display	Data display	Comment
- MEX	⌐		Memory examine mode
- 1000	1 0 0 0		Address of the first instruction
- NEX	1 0 0 0	X X	Content of this address=XX
- 3E	1 0 0 0	3 E	Change this content
- NEX	1 0 0 1	X X	Update and and automatic address increment
- 40	1 0 0 1	4 0	Next byte
- NEX, C6	1 0 0 2	C 6	
- NEX, OE	1 0 0 3	0 E	
- NEX, 32	1 0 0 4	3 2	
- NEX, 00	1 0 0 5	0 0	*The low address byte is always loaded before the high byte in an instruction.*
- NEX, 11	1 0 0 6	1 1	
- NEX, 76	1 0 0 7	7 6	
- NEX	1 0 0 8	X	Always depress the NEX key to update the last entry.

This program is now stored in the RAM and is located between memory addresses *1000H and 1007H*. *Let's now check if the* 1100H memory location contains the result of the addition.

Key	Address display	Data display	Comment
- MEX	⌐		
- 1100, NEX	1 1 0 0	X X	(1100)=X.

The content of the 1100H memory location is not equal to 4EH (40H+0EH=4EH). That's because the program has not yet been executed. Also, it is a good practice to check the program before executing it. After the program has been checked, set the PC to the address of the first instruction (1100H) then depress EXe :

- REX	r						Register examine
- PC(5), NEX	5				X	X	
- 10	5				1	0	Set the high byte
- NEX	5			-	X	X	Update and display the low byte
- 00, NEX	5				1	0	Update the low byte and display the high byte.
- EXe							Execution: the user's program takes the control, the display goes dark.

We should check the content of 1100H to make sure that the program has been properly executed.

- RES	⌐					
- 1100, NEX	1	1	0	0	4	E

We find that the content of the 1100H memory location is the sum of 40H and 0EH.

3.7. SINGLE STEP

This function is initiated by the SST key. When single stepping, the contents of the image registers are first reloaded into the Z-80 registers. The instruction is then executed and the monitor regains control enabling the contents of the Z-80 registers to be stored away once again in their image registers. The next address in the program, and the content of the accumulator are then displayed. Single stepping provides the user with an efficient method to debug the program under development.

3.7.1. APPLICATION

Using the same program, we will now proceed step by step. The purpose of this exercise is to change 40H to 35H:

Keys	Address display	Data display	Comment
- REX	r		Examine register mode
- PC, NEX	5	X X	
- 10, NEX	5 -	X X	Initialization of the program counter.
- 00, NEX	5	1 0	
- SST	1 0 0 2	4 0	The first instruction is executed; the monitor displays the address of the next instruction and the content of the accumulator (A-register)
- REX, A	A		
- NEX, 35	A	3 5	Change 40Hto 35H
- NEX	A A	X X	Acquisition; (A)=35
- SST	1 0 0 4	4 3	The second instruction is executed; the content of the accumulator is displayed (result of the addition: 35H + OEH = 43H); the PC points to the address of the next executable instruction.
- SST	1 0 0 7	4 3	The content of the accumulator is transfered to the 1100H memory location; the program counter points to the address of the last executable instruction (HALT).

Key	Address	Data	Comment
- MEX	⌐		Return to the MEX mode to check the content of 1100H memory location.
- 1100, NEX	1 1 0 0	4 3	(1100H)=43H
- SST	1 0 0 8	4 3	The accumulator has not executed any operation and its content remains the same ; the PC content is incremented by 1 to point to the next memory address (outside of program).

3.8. CASSETTE RECORDING

The PRO-80 offers a simple and inexpensive way to record on a cassette tape the volatile information stored in the RAM. Most audio cassette recorders can be used, however if you are planning to buy one for this purpose, we suggest you consider the 26-1206 model CTR-80 from Radio Shack. The transfer sequence is the following:

- Connect the "AUX" jack on the PRO-80 to the auxiliary input of the cassette recorder.

- Fully rewind the cassette

- Using the MEX mode enter the address of the last byte in the program to be recorded into the following memory locations:

  - 13DCH: Low address byte

  - 13DDH: High address byte

- Remember to depress the NEX key to validate the last entry.

- Turn the volume control to a half way setting.

- Initiate the cassette writefunction by depressing the CW key.

- Set the recorder in the record mode; the display unit will go dark.

- The " ⌐ " prompt sign reappears when the recording is complete.

3.9 CASSETTE READ

To transfer a program from a cassette to a RAM, simply follow these steps:

- Plug the CASS connector on your PRO-80 into the earphone (or monitor) jack on your recorder.

- Rewind the cassette and set the recorder in the play mode.

- Adjust the volume control so that the LED becomes very bright. (the volume control will most likely have to be set to maximum.)

- Depress the RES key on your PRO-80.

- Initiate the read function by depressing the CR push button; the display unit will go dark during the transfer.

- If the transfer is successful, the monitor returns to the MEX mode and displays the " ⌐¬ " prompt sign.

- If an error is detected during the transfer, the monitor displays the letter "r" in the first leftmost digit position. Check the volume and repeat the same procedure or check the program being read using the MEX mode and correct any error.

NOTE:

1. The more experienced reader will soon realize that it is not easy to carry out all PRO-80 functions with only a 1 Kbyte monitor. Because of such space limitations, the PRO-80 monitor can not record a program at just any RAM address. To be recorded all programs must be located at starting address 1000H.

2. The user can easily record and locate several programs on a single cassette thanks to the LED which is lit when data is present and off between recordings.

IV APPLICATION PROGRAMS

The reader must be familiar with the Z-80 set of instructions to understand the programs discussed in this chapter. Those who have had prior experience with machine language programming can refer directly to the "Micro-Reference Manual" which summarizes the Z-80 set of instructions. If you have no such experience consult : "Programming the Z-80" by Rodnay Zaks, Sybex, or any book dealing with the Z-80 instructions.

4.1. CHASER LIGHTS

The purpose of this simulation is to help the reader understand how the PRO-80 display unit works and how to write a program for a chaser light display.

Annex 1 represents the electronic diagram of the PRO-80 display unit. The functional illustration of the unit is in figure 4.1.

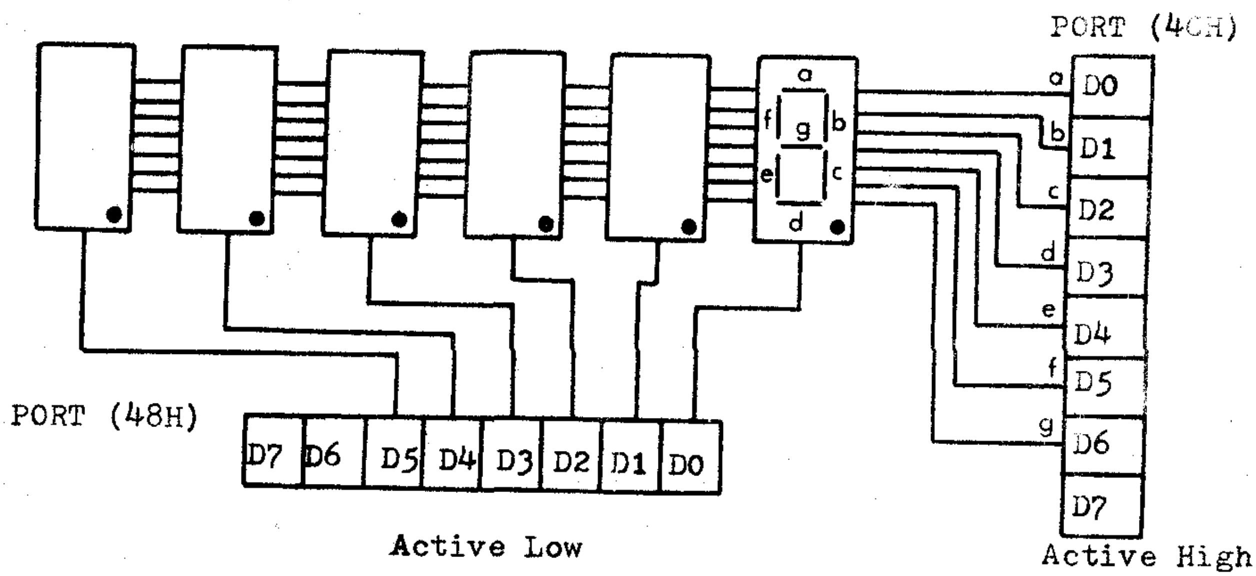


Figure 4.1: Functional diagram of the PRO-80 display unit.

This figure shows that each digit is made up of seven segments; a, b, c, d, e, f, and g. Each segment lights up when its corresponding bit at port 4CH is set to "1". The content of port 48H selects the active digit. For instance the first rightmost digit is active when the least significant bit is at logical "0", the second digit is active when the first order bit is at logical "0" and so on. If we wanted the first rightmost digit to be a 9, the content of the two ports would have to be:

Port 4CH	X	1	1	0	1	1	1	1	=(6FH)
Port 48H	X	X	1	1	1	1	1	0	=(3EH)

Exercise: We want the second rightmost digit to be a 3. Give the content of each port.

Answer: (Port 4CH) = 4FH
(Port 48H) = 3DH

To simulate chaser lights, we will be using segments c, d, e and g of each digit. If we want the "square" to move from right to left, the program sequence is the following:

1. Load the segment select word into the accumulator.

2. Transfer the content of the accumulator to port 4CH to activate the segments.

3. Load the digit select word into the accumulator.

4. Transfer the content of the accumulator.

5. Create a display delay.

6. Shift the content of the accumulator one position to the left to activate the next digit.

7. Loop to step 4 so the shift is repeated indefinitely.

X: This digit is not used by the display unit and can therefore be either "0" or "1". The "0" value has arbitrarily been given.

NOTE:

The display delay allows the operator to see the digits moving. Without it, the microcomputer operates at such speed that the all digits will appear to be always lit.

To create a delay we can for instance tell the processor to decrement a register already loaded with a value which determines the waiting time. When the content of the register becomes null, the processor continues to execute the remaining program instructions. If this delay is not long enough (as is presently the case), two or more registers may be used.

PROGRAM SEQUENCE:

1000	3EDC		LDA,DCH	Content of A=DCH
1002	D34C		OUT (4CH),A	Activate segments C,D,E,G.
1004	3EFE		LDA, FEH	(A)=FE
1006	D348	Loop 3:	OUT (48H),A	Select the 1st digit
1008	0E40		LDC, 40H	Initialize the first delay register.
100A	06FF	Loop 2:	LDB, FFH	Initialize the second delay register.
100C	10FE	Loop 1:	DJNZ, Loop1.	Decrement B to 0H
100E	0D		DEC C	Decrement C by 1.
100F	20F9		JRNZ,Loop 2:	Jump to loop 2 except if C=0H
1011	07		RLCA	Next digit.
1012	18F2		JR Loop 3:	Jump to loop 3.

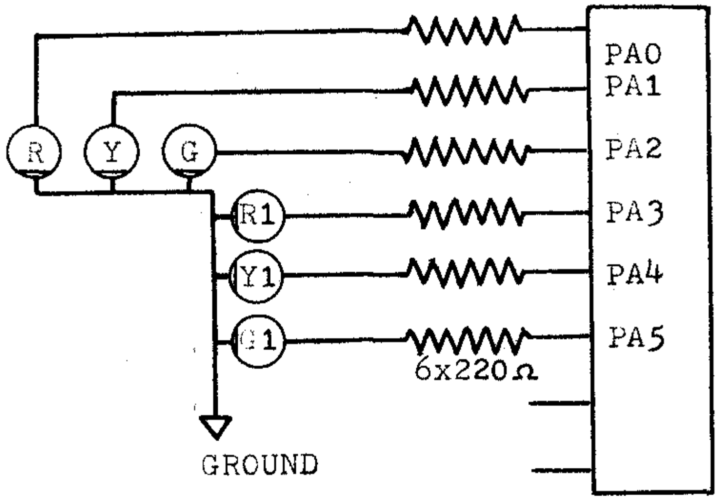
After studying this program, load it using the MEX mode. Find a simple way to:

1. Slow down the speed at which the square moves.

2. Replace the " □ " sign by another sign of your choice.

4.2. TRAFFIC LIGHTS

It would again be possible to use the display unit to simulate traffic lights but why not take this opportunity to try out your PRO-80 PIO. For this exercise, get two red, two yellow and two green LEDs[1] and six 220 Ω, ¼ W resistances to be assembled as follows:



Here is the instruction sequence:

a- Initialization: R and G1 are lit

b- Long delay

c- R and Y1 are lit

d- Short delay

e- R1 and G are lit

f- Long delay

g- R1 and Y are lit

h- Short delay

i- Loop to (a)

a- Initialization

Before the initialization begins, the PIO A-register must be set as an output port. Its control register (port 42H) should have the following content: (Consult your Micro-Reference Manual, page 25)

Port (42H)

0	0	X	X	1	1	1	1

(A control register).

(1) LED: Light-emitting diode.

The A-register is configured as an output port when D0 to D3 are at "1" and D6-D7 are at "0". Select either 0 or 1 for D4 and D5. The control word is 0000 1111 (OFH).

The A-register can now be initialized. R and G1 must be lit ; D0 and D5 are therefore at "1". All other lines are at "0" except for D6 and D7 which are ignored. The A-register (port 40H) will contain the following value:

Port A (40H)									
	0	0	1	0	0	0	0	1	(21H)

b- LONG DELAY

Since a delay will often be used, we suggest you to write a delay routine that could be called once for a short delay and n times for a long delay.

c- R AND YI ARE LIT

Data lines D0 and D4 must be set to "1", and all others must be at "0". The new control word transfered to port A is:

Port (40H)									
	0	0	0	1	0	0	0	1	(11H)

The word OCH is sent to port A to light R1 and G and the word OAH is sent to light R1 and Y.

PROGRAM

1000	3E OF		LDA, OFH	Configuration of the A register as an output port.
1002	D3 42		OUT(42H), A	
1004	3E 21	BR3:	LDA, 21H	
1006	D3 40		OUT (40H), A	R and G1 light up
1008	06 OA		LDB, OAH	
100A	CD 2A10	BR1:	CALL DELAY	Long Delay
100D	10 FB		DJNZ, BR1	
100F	3E 11		LDA, 11H	
1011	D3 40		OUT (40H),A	R and Y1 light up
1013	CD 2A10		CALL DELAY	Short delay
1016	3E OC		LDA, OCH	

```
1018   D3 40              OUT (40H),A   R1 and G light up
101A   06 0A              LDB, 0AH
101C   CD 2A10   BR2 :    CALL DELAY    LONG DELAY
101F   10 FB              DJNZ, BR2
1021   3E 0A              LDA, 0AH
1023   D3 40              OUT(40H),A
1025   CD 2A10            CALL DELAY    Short delay
1028   18 D9              JR,BR3        Loop

                       DELAY ROUTINE

102A   C5                 PUSH B        Save B and C
102B   0E 0B              LDC, 0B       Initialize the first counter
102D   06 FF     BR5 :    LDB, FF       Initialize the second counter
102F   10 FE     BR4 :    DJNZ, BR4     Decrement B to "0"
1031   0D                 DEC C         Decrement C
1032   20 F9              JRNZ, BR5     Test C=0
1034   C1                 POP B         Restore B and C
1035   C9                 RET           Return
```

4.3. DIGITAL CLOCK

We have seen in paragraph 4.1. how to activate a single digit of the display unit. Now, for several digits to appear lit at the same time, we must use the multiplexing technique. This technique consists in displaying sequentially one digit at a time and repeating the sequence fast enough so that the human eye cannot tell when the digits are not lit. Remember that in North America, alternating voltage shut off 120 times per second. For instance, the common neon tube is in fact shut off 120 times per second. It appears to be always lit because it is on for a longer time than it is off and at this frequency the retina is unable to tell when it is off and only senses the mean intensity of the light being emitted.

The digital clock simulation program is written in two steps ; in the first, a display routine is created which fetches the content of a buffer and converts it in order to generate and display the required number.

The second step consists in writing a program which will use the display routine to simulate a digital clock.

DISPLAY ROUTINE :

We must first of all define a look up table which will generate the control character required to light up each decimal digit (0 to 9). If for instance the starting address is 1200H, we would have the following table :

ADDRESS	CONTROL CHARACTER	DIGIT
1200H	3FH	0
1201H	06H	1
1202H	5BH	2
1203H	4FH	3
1204H	66H	4
1205H	6DH	5
1206H	7DH	6
1207H	07H	7
1208H	7FH	8
1209H	6FH	9

The next step would be to define six memory locations containing at any given moment a time value, especially the setting time.

(1210H)	=	seconds (units)	S1
(1211H)	=	seconds (tens)	S10
(1212H)	=	minutes (units)	M1
(1213H)	=	minutes (tens)	M10
(1214H)	=	hours (units)	H1
(1215H)	=	hours (tens)	H10

Now, we must follow the sequence below:

- Load the content of 1210H (S1) into a register.
- Find the control character in the look up table.
- Display it in the first rightmost digit position.
- Create a delay so that the digit remains displayed for a while.
- Clear this digit.
- Increment the starting address to select S10.
- Repeat the procedure until the six digits have been displayed.

CLOCK SIMULATION PROGRAM

The algorithm found on page 52 is a summary of the following program:

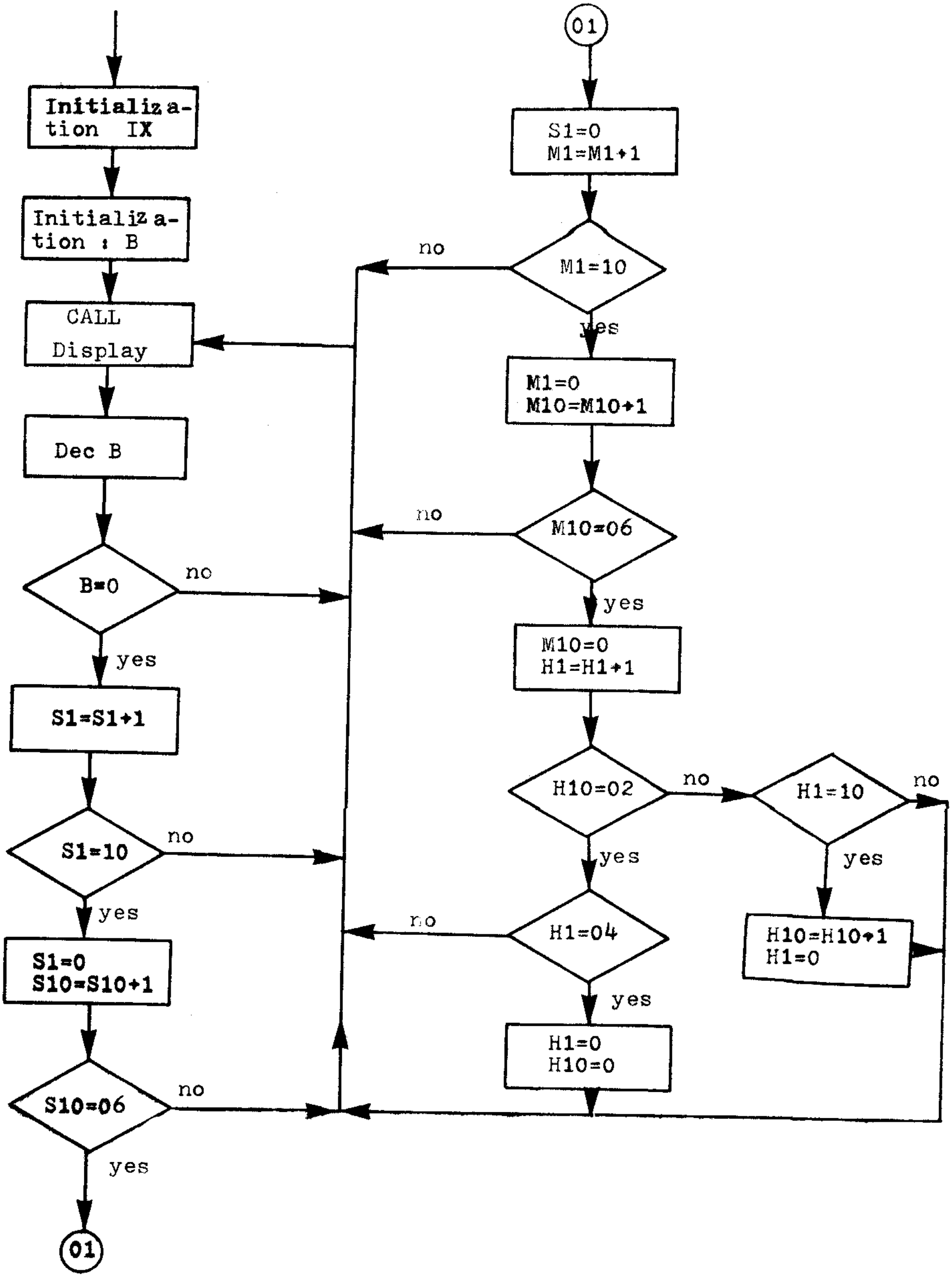
```
1000   DD211012          LDIX, 1210H          Pointer initialization
1004   0610              LDB, 10H             1 second initialization
1006   CD7310     BR1 :  CALL DISPLAY         Call display
1009   10FB              DJNZ, BR1            1s delay loop
100B   DD3400            INC (IX +0)          SEC1 increment
100E   3E0A              LDA, OAH
1010   DDBE00            CP (IX + 0)          TEST SEC 1=10
1013   20F1              JRNZ,BR1             Loop to display
1015   DD360000          LD(IX + 0),00H       SEC1=0
1019   DD3401            INC (IX+1)           SEC10 increment
101C   3E06              LDA,06H
101E   DDBE01            CP (IX + 1)          Test sec 10=06
1021   20E3              JRNZ,BR1             Loop to display
1023   DD360100          LD (IX + 1),00H      SEC 10=0
1027   DD3402            INC (IX + 2)         M1 increment
102A   3E0A              LDA,OAH
102C   DDBE02            CP (IX + 2)          TEST M1=10
102F   20D5              JRNZ,BR1             Loop to display
1031   DD360200          LD (IX + 2),00H      M1=0
1035   DD3403            INC (IX +3)          M10 increment
1038   3E06              LDA, 06H
103A   DDBE03            CP (IX + 3)          Test M10 = 06
103D   20D7              JRNZ,BR1             Loop to display
103F   DD360300          LD (IX + 3),00H      M10=0
```

```
1043  DD3404              INC (IX + 4)       H1 increment
1046  3E02                LDA,02H
1048  DDBE05              CP (IX + 5)        TEST H1=02
104B  2003                JRNZ, BR2
104D  C36110              JP, BR3            H1 = 02
1050  3E0A       BR2:     LDA, 0A
1052  DDBE04              CP (IX + 4)        TEST H1 = 10
1055  20AF                JRNZ,BR1           Loop to display
1057  DD360400            LD (IX + 4),00H    H1 = 0
105B  DD3405              INC (IX + 5)       H10 increment
105E  C30610              JP,BR1             Loop to display
1061  3E04       BR3:     LDA,04H
1063  DDBE04              CP (IX + 4)        TEST H10 = 04
1066  209E                JRNZ,BR1           LOOP to display
1068  DD360400            LD (IX + 4),00H    H1 = 0
106C  DD360500            LD (IX + 5),00H    H10 = 0
1070  C30610              JP, BR1

                          DISPLAY ROUTINE

1073  0EFE                LDC,FEH
1075  211012              LDHL, 1210H        Pointer initialization
1078  5E         BR5:     LDE, (HL)
1079  1612                LDD, 12            (DE) = CONTROL CARACTER
107B  79                  LDA,(C)
107C  D348                OUT (48H), A       Active digit
107E  1A                  LDA,(DE)
107F  D34C                OUT(4CH),A         Active segments
```

```
1081   3E4B           LDA, 4BH        Display delay
1083   3D      BR4:   DECA
1084   20FD           JRNZ,BR4
1086   D34C           OUT (4CH),A     Reset all segments
1088   23             INC HL          Next digit
1089   CB01           RLC, C
108B   CB71           BIT6, C         Test end of display
108D   20E9           JRNZ,BR5
108F   C9             RET             Return to program
```

BUFFER MEMORY

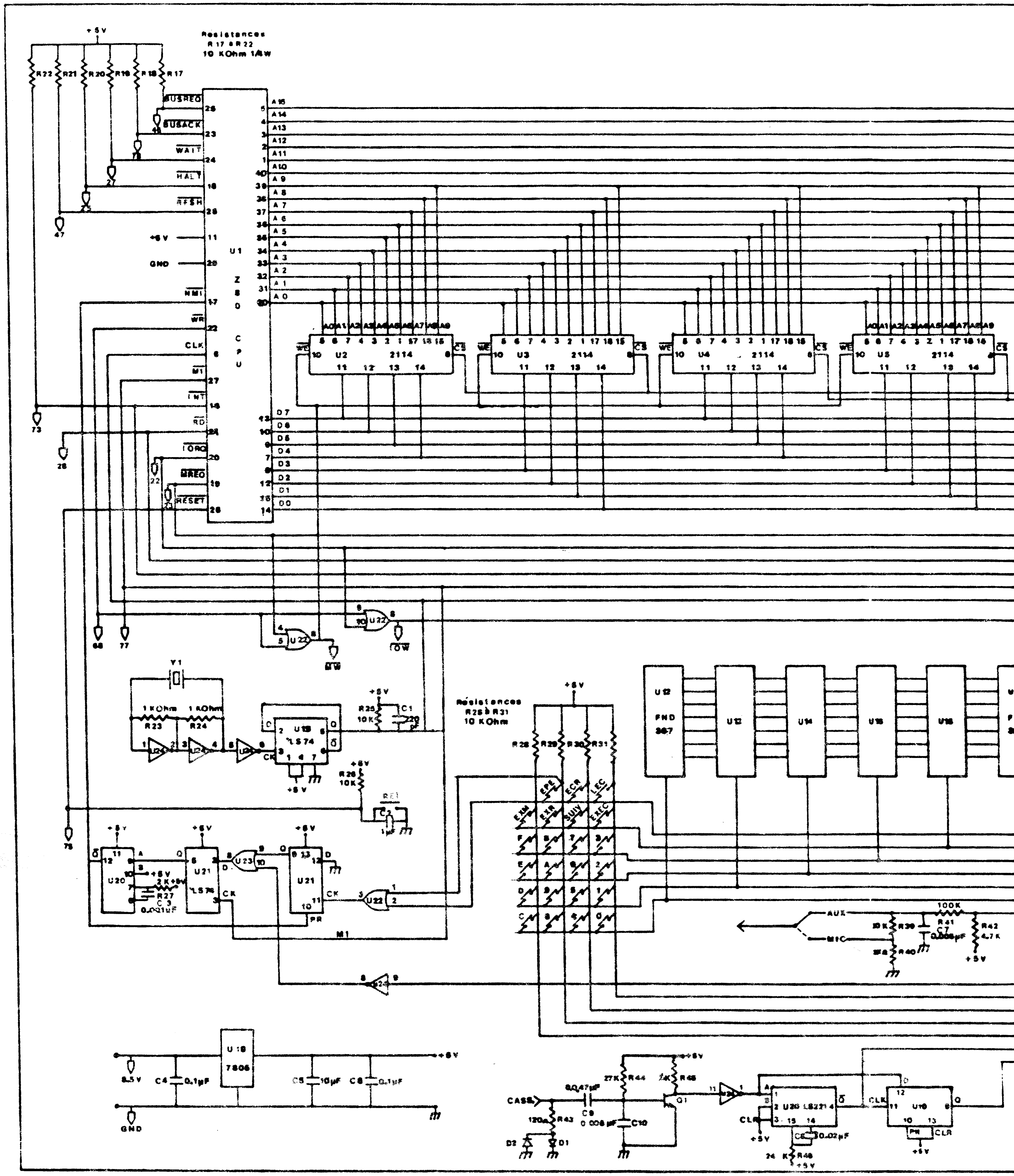
```
1200                  3F              7 segment digit control
                                      bytes (0-9)
1201                  06
1202                  5B
1203                  4F
1204                  66
1205                  6D
1206                  7D
1207                  07
1208                  7F
1209                  6F
1210                  SEC 1
1211                  SEC 10
1212                  MIN 1
1213                  MIN 10
1214                  H1
1215                  H10
```



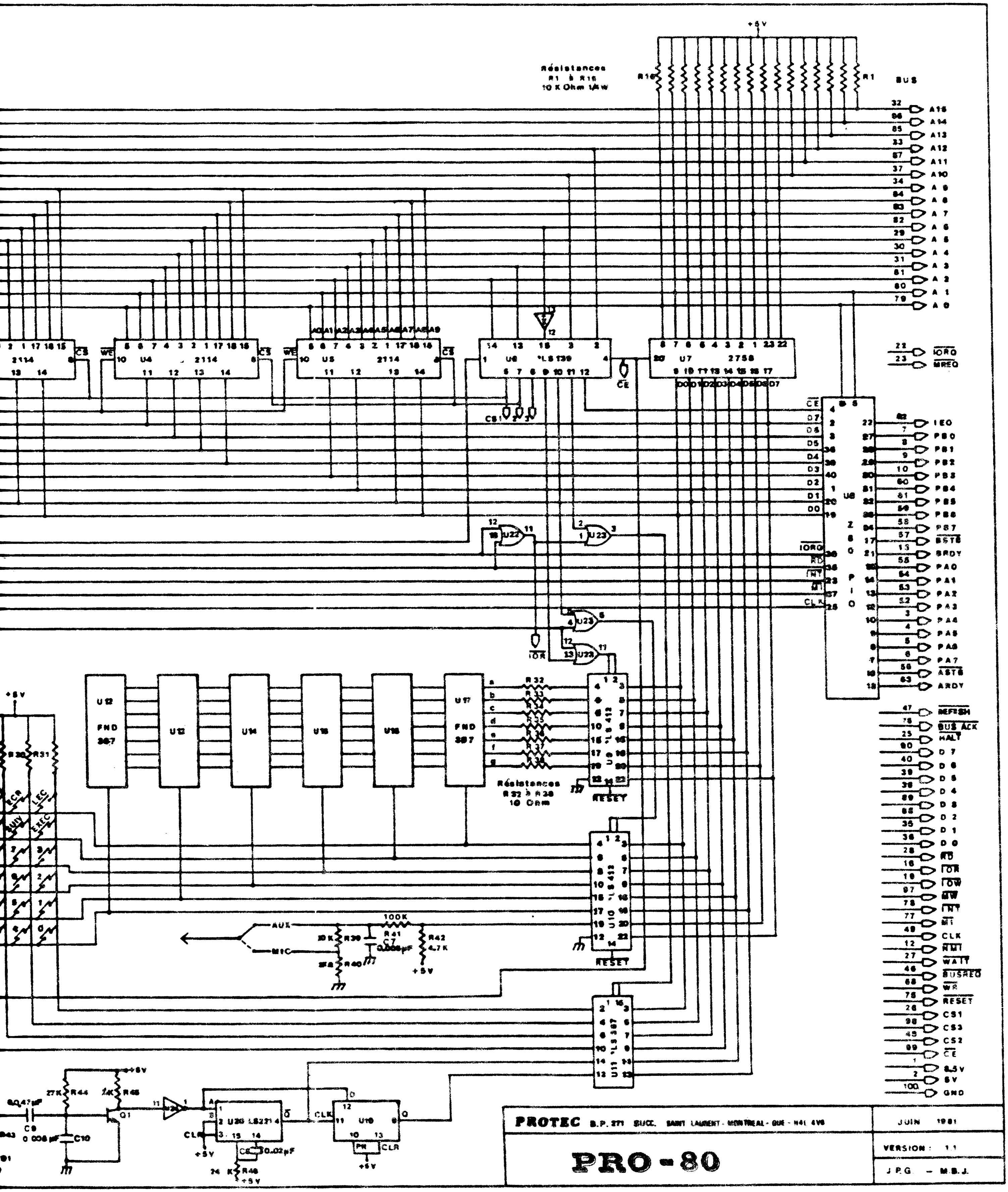
Initialization IX
Initialization : B
CALL Display
Dec B
B=0
no
yes
S1=S1+1
S1=10
no
yes
S1=0
S10=S10+1
S10=06
no
yes
01
01
S1=0
M1=M1+1
M1=10
no
yes
M1=0
M10=M10+1
M10=06
no
yes
M10=0
H1=H1+1
H10=02
no
yes
H1=10
no
yes
H10=H10+1
H1=0
H1=04
no
yes
H1=0
H10=0

ANNEX 1 :

PRO-80 DIAGRAM



Resistances R 17 à R 22 10 KOhm 1/4W
Resistances R 28 à R 31 10 KOhm
U1 Z 80 CPU
2114
FND 357
7805



Résistances R1 à R16 10 K Ohm 1/4 W
BUS
A15
A14
A13
A12
A11
A10
A 9
A 8
A 7
A 6
A 5
A 4
A 3
A 2
A 1
A 0
2114
U4
U5
U6
'LS 139
U7
2758
IORQ
MREQ
CE
CS1
U8
Z80 PIO
IEO
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
BSTB
BRDY
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
ASTB
ARDY
U22
U23
IOR
U12 FND 357
U13
U14
U15
U16
U17 FND 357
Résistances R32 à R38 10 Ohm
RESET
ECR
LEC
RUN
EXEC
AUX
MIC
100 K
R41
R42 4.7 K
+5 V
REFRSH
BUS ACK
HALT
D 7
D 6
D 5
D 4
D 3
D 2
D 1
D 0
RD
IOR
IOW
MW
INT
M1
CLK
NMI
WAIT
BUSREQ
WR
RESET
CS1
CS3
CS2
CE
8.5 V
5 V
GND
U20 LS221
U19
Q1
PROTEC B.P. 271 SUCC. SAINT LAURENT - MONTREAL - QUE - H4L 4V6
JUIN 1981
PRO-80
VERSION : 1.1
J.P.G - M.B.J.

ANNEX II

PRO-80 MONITOR

RAM MEMORY MAP LIST.

1386		
-		USER STACK POINTER
13A0		
13A1		
-		MONITOR STACK POINTER
13BA		
13BB	RST8	
13BE	RST16	
13C1	RST24	VECTOR ADDRESSES FOR "RESTART" INSTRUCTIONS RST 8 TO RST 56
13C4	RST32	
13C7	RST40	
13CA	RST48	
13CD	RST56	
13D0	RTD1	
13D1	RTD2	
13D2	RTD3	BUFFER REGISTERS: HEX VALUE OF DIGITS TO BE DISPLAYED
13D3	RTD4	
13D4	RTD5	
13D5	RTD6	
13DC	OCARIN	LAST ADDRESS OF PROGRAM TO BE RECORDED (LOW BYTE)
13DD	OCASU	LAST ADDRESS OF PROGRAM TO BE RECORDED (HIGH BYTE)
13DE	TOUCOU	CURRENT KEY
13DF	REGRANG	ROW REGISTER
13E0	POINDIG	DIGIT POINTER
13E1	MTD1	
13E2	MTD2	
13E3	MTD3	BUFFERS FOR DISPLAY DIGITS.
13E4	MTD4	CONTAIN 7 SEGMENT CONTROL BYTES.
13E5	MTD5	
13E6	MTD6	
13E7	POINREG	FLAG REGISTER

13E8	RIMPPH		IMAGE REGISTER STACK POINTER "HIGH BYTE"
13E9	RIMCOH		IMAGE REGISTER PROGRAM COUNTER "HIGH BYTE"
13EA	RIMIXH	IX	IMAGE REGISTER "HIGH BYTE"
13EB	RIMIYH	IY	IMAGE REGISTER "HIGH BYTE"
13EC	RIML	L	IMAGE REGISTER
13ED	RIMH	H	IMAGE REGISTER
13EE	RIMA	A	IMAGE REGISTER
13EF	RIMB	B	IMAGE REGISTER
13F0	RIMC	C	IMAGE REGISTER
13F1	RIMD	D	IMAGE REGISTER
13F2	RIME	E	IMAGE REGISTER
13F3	RIMF	F	IMAGE REGISTER
13F4	RIMPPB		IMAGE REGISTER STACK POINTER "LOW BYTE"
13F5	RIMCOB		IMAGE REGISTER PROGRAM COUNTER "LOW BYTE"
13F6	RIMIXB	IX	IMAGE REGISTER "LOW BYTE"
13F7	RIMIYB	IY	IMAGE REGISTER "LOW BYTE"
13F8	RIML'	L'	IMAGE REGISTER
13F9	RIMH'	H'	IMAGE REGISTER
13FA	RIMA'	A'	IMAGE REGISTER
13FB	RIMB'	B'	IMAGE REGISTER
13BC	RIMC'	C'	IMAGE REGISTER
13FD	RIMD'	D'	IMAGE REGISTER
13FE	RIME'	E'	IMAGE REGISTER
13FF	RIMF'	F'	IMAGE REGISTER

PRO-80 MONITOR ,ALL RIGHTS RESERVED © 1981 PROTEC

** INITIALIZATION **

```
0000  31BA13           LD SP,13BA        MONITOR STACK POINTER = 13BA
0003  21D013           LD HL,RTD1        HL POINTS TO FIRST DIGIT BUFFER REGISTER
0006  1803             JR BRO
0008  C3BB13           JP RST8           JUMP TO RST 8 RAM ADDRESS
000B  0630     BRO:    LD B,30
000D  00               NOP
000E  1803             JR BR1
0010  C3BE13           JP RST16          JUMP TO RST 16 RAM ADDRESS
0013  3E00     BR1;    LD A,00
0015  77       BR2;    LD(HL),A          RESET LOCATION POINTED BY HL
0016  1803             JR BR3
0018  C3C113           JP RST24          JUMP TO RST 24 RAM LOCATION
001B  23       BR3:    INC HL            NEXT ADDRESS
001C  10F7             DJNZ,BR2          LOOP
001E  1803             JR BR4
0020  C3C413           JP RST 32         JUMP TO RST 32 RAM LOCATION
0023  3E13     BR4:    LD A, 13
0025  00               NOP
0026  1803             JR BR5
0028  C3C713           JP RST 40         JUMP TO RST 40 RAM LOCATION
002B  32E813   BR5:    LD(RIMPPH), A     IMAGE REGISTER PPH= 13
002E  1803             JR BR6
0030  C3CA13           JP RST48          JUMP TO RST48 RAM LOCATION
0033  3EA0     BR6:    LD A,A0
0035  00               NOP
0036  1803             JR BR7
0038  C3CD13           JP RST56          JUMP TO RST 56 RAM LOCATION
003B  32F413   BR7:    LD(RIMPPB),A      PPB=A0; END OF INITIALIZATION
```

** MEX **
EXAMINE AND CHANGE THE CONTENT OF MEMORY LOCATIONS

```
003E  CD7600   MEX:    CALL MTBL         RESET SIX MEMORY DIGIT BUFFERS
0041  3E54             LD A,54           MTD6=
0043  32E613           LD(MTD6),A
0046  06DF             LD B,DF           DIGIT POINTER = 6
0048  0E00             LD C,00           ADRESS MODE    = 00
004A  51               LD D,C            ADDRESS FLAG   = 00
004B  59               LD E,C            NEXT MODE      = 00
004C  CD8700   MEX1:   CALL LECDEC       READ AND DECODE KEYBOARD
004F  79               LD A,C
0050  C600             CP A,00           TEST ADDRESS MODE = 00
0052  200E             JRNZ,BR8          JUMP TO ADDRESS MODE = 01
0054  CB50             BIT 2,B           TEST DIGIT POINTER   = DIG.3
0056  2002             JRNZ,02
0058  1601             LD D,01           FLAG ADDRESS = 01
005A  CB48             BIT 1,B           TEST DIGIT POINTER = DIG.2
005C  2002             JRNZ,02
005E  06DF             LD B,DF           DIGIT POINTER = DIG.6
0060  180B             JR BR9
0062  CB78     BR8:    BIT 7,B           TEST TWO DIGITS DISPLAYED
0064  1803             JR BR10
```

```
0066  C3A202          JP SST          JUMP TO SINGLE STEP ROUTINE
0069  2002     BR10:  JRNZ,BR9
006B  06FD            LD B,FD         DIGIT POINTER = DIG.2
006D  3E00     BR9:   LD A,00
006F  CD 3D01         CALL CHART      ACTIVE KEY IN BUFFER REGISTER AND BUFFER MEMORY
0072  CB08            RRC B           NEXT DIGIT
0074  18D6            JR MEX1         JUMP TO SCAN AND DECODE
```

** MTBL **

BUFFER DIGIT CONTENT IS RESET

```
0076  F5       MTBL:  PUSH A          SAVE REGISTERS
0077  E5              PUSH H
0078  0606            LD B,06
007A  3E00            LD A,00
007C  21E613          LD HL,MTD6      INITIALIZATION-BUFFER POINTER
007F  77       BR11:  LD (HL),A       BUFFER MEMORY = 00
0080  2D              DEC L           NEXT BUFFER
0081  05              DEC B
0082  20FB            JRNZ,BR11       TEST END OF RESET
0084  E1              POP H           RESTORE REGISTERS
0085  F1              POP A
0086  C9              RET
```

** LECDEC **

READ AND DECODE KEYBOARD

```
0087  C5       LECDEC:PUSH B          SAVE REGISTERS USED
0088  E5              PUSH H
0089  21E613   BR12:  LD HL,MTD6      INITIALIZATION-BUFFER POINTER
008C  3EDF            LD A,DF
008E  32E013   BR13:  LD(POINDIG),A   INITIALIZATION-DIGIT POINTER
0091  0E00            LD C,00         FLAG KEY = 0 (NON ACTIVE)
0093  D348            OUT(DIG),A      ACTIVE DIGIT
0095  DB44     BR14:  IN(CLAV),A      READ COLUMN
0097  00              NOP
0098  E60F            AND OF          BITS 4 TO 7 MASKED
009A  CB41            BIT 0,C         TEST FLAG KEY = 1
009C  2003            JRNZ,BR15
009E  32DF13          LD(REGRANG),A   TRANSFER TO ROW REGISTER
00A1  FEOF     BR15:  CP A,OF         TEST KEY DEPRESSED
00A3  280B            JRZ,BR16
00A5  0E01            LD C,01         FLAG KEY = 1
00A7  0620            LD B,10
00A9  CD3301   BR18:  CALL DEL        DEBOUNCE DELAY
00AC  10 FB           DJNZ, BR18
00AE  18E5            JR BR14         JUMP TO TEST IF KEY STILL ACTIVE
00B0  CB41     BR16:  BIT 0,C
00B2  2015            JRNZ,BR17       JUMP TO DECODE
00B4  7E              LD A,(HL)       CURRENT DIGIT DISPLAY
00B5  D34C            OUT(SEG),A
```

```
00B7  CD3301           CALL DEL          CALL DISPLAY DELAY
00BA  3E00             LD A,00           CLEAR DISPLAY
00BC  D34C             OUT(SEG),A
00BE  2D               DEC L             NEXT BUFFER
00BF  3AE013           LD A,(POINDIG)
00C2  CB47             BIT 0,A           TEST END OF DISPLAY
00C4  28C3             JRZ,BR12
00C6  0F               RRCA              NEXT DIGIT
00C7  18C5             JR BR13           JUMP TO KB SCAN
00C9  E1        BR17:  POP H             RESTORE REGISTERS
00CA  C1               POP B
00CB  33               INC SP
00CC  33               INC SP
00CD  3AE013           LD A, (POINDIG)
00D0  CB47             BIT 0,A           TEST DIGIT POINTER = DIG.1
00D2  2012             JRNZ, BR20
00D4  3ADF13           LD A,(REGRANG)
00D7  CB47             BIT 0,A           TEST ROW POINTER = ROW.1
00D9  2003             JRNZ,BR19
00DB  C38303           JP CARE           JUMP TO CASSETTE TAPE READ
00DE  CB4F      BR19:  BIT 1,A           TEST SECOND ROW
00E0  C25402           JPNZ,EXEC         JUMP TO SINGLE STEP
00E3  C31303           JP CAW            JUMP TO CASSETTE WRITE
00E6  CB4F      BR20:  BIT 1,A           TEST SECOND COLUMN
00E8  2019             JRNZ,BR23
00EA  3ADF13           LD A,(REGRANG)
00ED  CB47             BIT 0,A           TEST FIRST ROW
00EF  2003             JRNZ,BR21
00F1  C35402           JP EXEC           JUMP TO EXECUTE ROUTINE
00F4  CB4F      BR21:  BIT 1,A           TEST SECOND ROW
00F6  2003             JRNZ,BR22
00F8  C37401           JP NEXT           JUMP TO "NEXT" ROUTINE
00FB  CB57      BR22:  BIT 2,A           TEST THIRD ROW
00FD  C23E00           JPNZ,MEX          JUMP TO MEMORY EXAMINE
0100  C3D001           JP REX            JUMP TO REGISTER EXAMINE
0103  3B        BR23:  DEC SP
0104  3B               DEC SP
0105  C5               PUSH B
0106  CD2401           CALL CONV         A= PLACE OF ZERO IN DIG POIN
0109  CB07             RLC A
010B  CB07             RLC A             MULTIPLY BY 4
010D  F5               PUSH A
010E  3ADF13           LD A,(REGRANG)
0111  CD2401           CALL CONV         A= PLACE OF ZERO IN ROWREG
0114  47               LD B,A
0115  F1               POP A
0116  80               ADDA,B            GENERATE NEW POINTER
0117  C6D8             ADD A,D8
0119  E5               PUSH H
011A  6F               LD L,A
011B  2603             LD H,03
011D  7E               LD A,(HL)         TRANSCODING
```

```
011E  E1                POP H
011F  32DE13            LD(TOUCOU),A    TRANSFER TO TOUCOU
0122  C1                POP B
0123  C9                RET
0124  C5        CONV:   PUSH B          UPON RETURN,THE A-REGISTER CONTAINS
0125  47                LD B,A          A BIT REPRESENTING THE PLACE
0126  3E00              LD A,00         OF ZERO IN THE BYTE IN DIG POIN
0128  CB40      BR24:   BIT 0,B
012A  2002              JRNZ,BR25
012C  C1                POP B
012D  C9                RET
012E  CB08      BR25:   RRC B
0130  3C                INC A
0131  18F5              JR BR24
```

** DEL **

DELAY ROUTINE

```
0133  C5        DEL:  PUSH B
0134  06F8            Ld B,F8
0136  10FE            DJNZ,FE
0138  05              DEC B
0139  10FE            DJNZ,FE
013B  C1              POP B
013C  C9              RET
```

** CHART **

THE CONTENT OF THE CURRENT KEY IS TRANSCODED
AND TRANSFERED TO THE CORRESPONDING BUFFER MEMORY

```
013D  D5        CHART: PUSH D           SAVE REGISTER
013E  E5               PUSH H
013F  F5               PUSH A
0140  21D013    BR26:  LD HL,RTD1       INITIALIZATION-BUFFER REGISTER POINTER
0143  78               LD A,B
0144  CD2401           CALL CONV
0147  85               ADD A,L
0148  6F               LD L,A           GENERATE NEW POINTER
0149  3ADE13           LD A,(TOUCOU)
014C  E60F             AND OF           MASK HIGH BYTE
014E  77               LD(HL),A         TRANSFER CURRENT KEY TO BUFFER REGISTER
014F  C6F0             ADD A,FO         CREATE SEGMENT CODE POINTER
0151  5F               LD E,A
0152  1603             LD D,03
0154  3E11             LD A,11          GENERATE BUFFER POINTER
0156  85               ADD A,L
0157  6F               LD L,A
0158  1A               LD A,(DE)
0159  77               LD(HL),A         TRANSFER SEGMENT CODE TO BUFFER MEMORY
015A  F1               POP A
015B  FE00             CP A,00          TEST 1 OR 2 DIGIT DISPLAY
015D  2812             JRZ,BR27
015F  3D               DEC A
0160  F5               PUSH A
0161  3ADE13           LD A,(TOUCOU)
```

```
0164  E6F0           AND F0          MASK HIGH BYTE
0166  0F             RRC A
0167  0F             RRC A
0168  0F             RRC A
0169  0F             RRC A
016A  32DE13         LD(TOUCOU),A    TRANSFER TO CURRENT KEY
016D  CB00           RLC B           NEXT DIGIT
016F  18CF           JR BR26
0171  E1      BR27:  POP H           RESTORE REGISTERS
0172  D1             POP D
0173  C9             RET
```

**** NEXT ****

DISPLAY AND CHANGE MEMORY LOCATION CONTENT

```
0174  CB43    NEXT:  BIT 0,E         TEST REX MODE
0176  C20202         JPNZ,NEXT 1
0179  CB42           BIT 0,D         TEST ALL ADDRESS DIGITS DISPLAYED
017B  CA4C00         JPZ,MEX1
017E  DD21D413       LD IX,RTD5      RTD POINTER INITIALIZATION
0182  CDC101         CALL LECDON     READ RTD 5, RTD 6
0185  67             LD H,A          TRANSFER TO H
0186  DD21D213       LD IX,RTD3
018A  CDC101         CALL LECDON     READ RTD 3, RTD 4
018D  6F             LD L,A          TRANSFER TO L
018E  06FE           LD B,FE         DIGIT POINTER INITIALIZATION
0190  CB41           BIT 0,C         TEST ADDRESS MODE
0192  281F           JRZ,BR30
0194  DD21D013       LD IX,RTD1
0198  CDC101         CALL LECDON     READ RTD 1, RTD 2
019B  77             LD(HL),A        TRANSFER DATA TO CURRENT LOCATION
019C  23             INC HL          NEXT ADDRESS
019D  06FB           LD B,FB         DIGIT POINTER = DIG.3
019F  7D             LD A,L
01A0  32DE13  BR28:  LD(TOUCOU),A    TRANSFER ADDRESS BYTE TO TOUCOU
01A3  3E01           LD A,01
01A5  CD3D01         CALL CHART      ADDRESS BYTE IN BUFFER REGISTER AND BUFFER MEMORY
01A8  CB00           RLC B
01AA  CB70           BIT 6,B         TEST END OF ADDRESS DISPLAY
01AC  2803           JRZ,BR29
01AE  7C             LD A,H          TRANSFER HIGH BYTE
01AF  18EF           JR BR28
01B1  06FE    BR29:  LD B,FE         DIGIT POINTER = DIG. 1
01B3  7E      BR30:  LD A,(HL)
01B4  32DE13         LD(TOUCOU),A    TRANSFER MEMORY LOCATION CONTENT TO
01B7  3E01           LD A,01         THE DATA REGISTER
01B9  CD3D01         CALL CHART
01BC  0E01           LD C,01         DATA MODE INITIALIZATION
01BE  C34C00         JP MEX1
```

** LECDON **

GENERATE A SINGLE BYTE TRANSFERED TO THE ACCUMULATOR

```
01C1  E5       LECDON: PUSH H
01C2  DD7E01           LD A,(IX/01)  READ FIRST NIBLE
01C5  OF               RRC A
01C6  OF               RRC A         RIGHT SHIFT
01C7  OF               RRC A
01C8  OF               RRC A
01C9  67               LD H,A
01CA  DD7E00           LD A,(IX/00)  READ SECOND NIBLE
01CD  B4               OR H          GENERATE BYTE
o1CE  E1               POP H
01CF  C9               RET
```

** REX **

READ AND CHANGE CONTENT OF REGISTERS

```
01D0  CD7600   REX:  CALL MTBL       RESET ALL SIX MEMORY BUFFERS
01D3  3E50           LD A,50
01D5  32E613         LD(MTD6),A      BUFFER MEMORY 6 = r
01D8  06DF           LD B,DF         DIGIT POINTER = DIG.6
01DA  1E01           LD E,01         NEXT MODE = 1
01DC  0E00           LD C,00         RESET REGISTER MODE
01DE  1655           LD D,55         ALTERNATE POINTER INITIALIZATION
01E0  CD8700   BR31: CALL LECDEC     READ DECODE
01E3  CB41           BIT 0,C         TEST REGISTER MODE
01E5  280A           JRZ,BR32
01E7  CB08           RRC B           NEXT REGISTER
01E9  CB78           BIT 7,B         TEST 2 DATA DIGITS DISPLAYED
01EB  2002           JRNZ,02
01ED  06FD           LD B, FD        DIGIT POINTER REINITIALIZATION
01EF  180A           JR BR33
01F1  3ADF13   BR32: LD A,(REGRANG)
01F4  FE0E           CP A,0E         TEST ACTIVE REGISTER
01F6  28E8           JRZ, BR31       NON EXISTANT REGISTER: RETURN TO SCAN
01F8  32E713         LD(POINREG),A   ACTIVE REGISTER IN REGISTER POINTER
01FB  3E00     BR33: LD A,00
01FD  CD3D01         CALL CHART      ACTIVE REGISTER IN BUFFER REGISTER AND
0200  18DE           JR BR31                                BUFFER MEMORY.
```

** NEXT 1 **

DISPLAY AND CHANGE REGISTER CONTENTS

```
0202  3AE613   NEXT1:LD A,(MTD6)
0205  FE50           CP A,50         TEST DISPLAYED REGISTER
0207  CAE001         JPZ,BR31
020A  CB41           BIT 0,C         TEST REGISTER MODE
020C  2827           JRZ,BR36
020E  DD21D013       LD IX,RTD1      POINTER REGISTER INITIALIZATION
0212  CDC101         CALL LECDON     DATA READ
```

```
0215  77                LD(HL),A          TRANSFER DATA TO IMAGE REGISTER
0216  CB02              RLC D
0218  CB42              BIT 0,D
021A  2806              JRZ,BR34          TEST ALTERNATE REGISTER
021C  7D                LD A,L
021D  D60C              SUB A, 0C         GENERATE REGISTER POINTER
021F  6F                LD L,A
0220  181B              JR BR37
0222  3E0C      BR34:   LD A,0C           GENERATE ALTERNATE REGISTER POINTER
0224  85                ADD A,L
0225  6F                LD L, A
0226  3AE713            LD A,(POINREG)    TEST 16 BIT REGISTER
0229  CB4F              BIT 1,A
022B  2804              JRZ,BR35
022D  3E77              LD A,77           TRANSFER "A" TO THIRD DIGIT
022F  180E              JR BR38
0231  3E40      BR35:   LD A,40           TRANSFER "_" TO THIRD DIGIT
0233  180A              JR BR38
0235  3ADE13    BR36:   LD A,(TOUCOU)     GENERATE IMAGE REGISTER POINTER
0238  C6E4              ADD A,E4
023A  6F                LD L,A
023B  2613              LD H,13
023D  3E00      BR37:   LD A,00           TRANSFER "BLANK" TO THIRD DIGIT
023F  32E313    BR38:   LD(MTD3),A
0242  06FE              LD B,FE           DIGIT POINTER = DIG.1
0244  7E                LD A,(HL)
0245  32DE13            LD(TOUCOU),A      IMAGE REGISTER IN CURRENT KEY
0248  3E01              LD A,01
024A  CD3D01            CALL CHART        TRANSFER IMAGE REGISTER TO BUFFER
024D  CB08              RRC B             NEXT DIGIT
024F  0E01              LD C,01           REGISTER MODE INITIALIZATION
0251  C3E001            JP BR31
```

** EXEC **

PROGRAM EXECUTE

```
0254  0604      EXEC:   LD B,04
0256  DD21E813          LD IX,RIMPPH      IX INITIALIZATION
025A  DD6600    BR39:   LD H,(IX+00)      IMAGE REGISTERS IN HL
025D  DD6E0C            LD L,(IX+0C)
0260  CB50              BIT 2,B
0262  2803              JRZ,BR40
0264  F9                LD SP,HL          TRANSFER USER STACK POINTER
0265  1801              JR BR41
0267  E5        BR40:   PUSH H            SAVE 16 BIT IMAGE REGISTER
0268  DD23      BR41:   INC IX            NEXT IMAGE REGISTER
026A  10EE              DJNZ,BR39         LOOP
026C  DD6601    BR42:   LD H,(IX+01)      HIGH BYTE IM.REG IN H
062F  DD6E00            LD L,(IX+00)      LOW BYTE IM.REG IN L
0272  E5                PUSH H            SAVE HL
0273  DD6602            LD H,(IX+02)      IMRA IN H
0276  DD6E07            LD L,(IX+07)      IMRF IN L
0279  E5                PUSH H            SAVE AF
027A  DD6503            LD H,(IX+03)
```

```
027D  DD6E04            LD L,(IX+04)
0280  E5                PUSH H          SAVE BC
0281  DD6605            LD H,(IX+05)
0284  DD6E06            LD L,(IX+06)
0287  E5                PUSH H          SAVE DE
0288  04                INC B
0289  CB48              BIT 1,B         TEST END OF SAVE REGISTERS
028B  2006              JRNZ,BR43
028D  DD21F813          LD IX,RIML'
0291  18D9              JR BR42
0293  D1         BR43:  POP D           RESTORE ALTERNATE REGISTERS
0294  C1                POP B
0295  F1                POP A
0296  E1                POP H
0297  08                EX AF,AF'
0298  D9                EXX
0299  D1                POP D           RESTORE OTHER REGISTERS
029A  C1                POP B
029B  F1                POP A
029C  E1                POP H
029D  FDE1              POP IY
029F  DDE1              POP IX
02A1  C9                RET             RESTORE PROGRAM COUNTER
```

** SST **

SINGLE STEP EXECUTE

```
02A2  22EC13     SST:   LD (RIML),HL    H AND L EACH IN THEIR IMAGE REGISTER
02A5  21EE13            LD HL,RIMA
02A8  77         BR44:  LD(HL),A        ACCUMULATOR IN ITS IMAGE REGISTER
02A9  23                INC HL
02AA  70                LD(HL),B        B IN ITS IMAGE REGISTER
02AB  23                INC HL
02AC  71                LD(HL),C        C IN ITS IMAGE REGISTER
02AD  23                INC HL
02AE  72                LD(HL),D        D IN ITS IMAGE REGISTER
02AF  23                INC HL
02B0  73                LD(HL),E        E IN ITS IMAGE REGISTER
02B1  23                INC HL
02B2  F5                PUSH A
02B3  D1                POP D
02B4  73                LD(HL),E        F IN ITS IMAGE REGISTER
02B5  7D                LD A,L
02B6  FEF3              CP A,F3         TEST END OF 8 BIT REGISTER TRANSFER
02B8  200A              JRNZ,BR45       JUMP TO END OF 8 BIT REGISTER TRANSFER
02BA  08                EX AF,AF'
02BB  D9                EXX             ALTERNATE REGISTERS
02BC  22F813            LD(RIML')HL     H' L' IN IMAGE REGISTER
02BF  21FA13            LD HL,RIMA'     H' L' POINT TO A' IMAGE REGISTER
02C2  18E4              JR BR44         LOOP TO ALTERNATE REGISTER TRANSFER
02C4  FDE5       BR45:  PUSH IY         SAVE IY
02C6  FD21E813          LD IY,RIMPPH    IY POINTS TO PPH IMAGE REGISTER
02CA  E1                POP H           TRANSFER IY TO HL
02CB  FD7403            LD(IY+03),L     IY "H" IN ITS IMAGE REGISTER
02CE  FD750F            LD(IY+0F),L     IY "L" IN ITS IMAGE REGISTER
02D1  DDE5              PUSH IX
02D3  E1                POP H           TRANSFER IX TO HL
02D4  FD7402            LD(IY+02),H     IX IN ITS IMAGE REGISTER
02D7  FD750E            LD(IY+0E),L
```

```
02DA  E1              POP H           TRANSFER PROGRAM COUNTER TO HL
02DB  FD7401          LD(IY+01),H
02DE  FD750D          LD(IY+0D),L     TRANSFER PC TO IMAGE REGISTER
02E1  210000          LD HL,0000
02E4  39              ADD HL,SP       TRANSFER SP TO HL
02E5  FD7400          LD(IY+00),H     STACK POINTER IN ITS IMAGE REGISTER
02E8  FD750C          LD(IY+0C),L
02EB  08              EX AF,AF'
02EC  D9              EXX             ALTERNATE REGISTERS
02ED  06FE            LD B,FE         DIGIT POINTER = DIG.1
02EF  3AEE13          LD A,(RIMA)     A IMAGE REGISTER IN A
02F2  32DE13    BR46: LD(TOUCOU),A
02F5  3E01            LD A,01         TWO DIGIT TRANSFER MODE
02F7  CD3D01          CALL CHART      TRANSFER A TO BUFFER REGISTER AND BUFFER MEMORY
02FA  CB00            RLC B           NEXT DIGIT
02FC  CB50            BIT 2,B         TEST PC "L" LOADED
02FE  2005            JRNZ,BR47
0300  3AF513          LD A,(RIMCOB)   PC "L" IN A
0303  18ED            JR BR46
0305  CB60      BR47: BIT 4,B
0307  2005            JRNZ,BR48       TEST PC "H" LOADED
0309  3AE913          LD A,(RIMCOH)
030C  18E4            JR BR46         PC "H" IN A
030E  CD8700    BR48: CALL LECDEC     READ DECODE
0311  18FB            JR BR48
```

** CAW **

CASSETTE TAPE WRITE

```
0313  1E00      CAW:  LD E,00         CHECKSUM REGISTER = 0
0315  06FF            LD B,FF         COUNTER INITIALIZATION FOR TRANSMISSION
0317  3E00      BR49: LD A,00         OF ZEROES
0319  CD4A03          CALL TRANSM
031C  10F9            DJNZ,BR49       TEST END OF "0" TRANSMIT
031E  2ADC13          LD HL,(OCARIN)  RECORD LAST LOCATION IN PROGRAM
0321  7C              LD A,H
0322  CD4A03          CALL TRANSM
0325  7D              LD A,L
0326  CD4A03          CALL TRANSM
0329  23              INC HL
032A  010010          LD BC,1000      START ADDRESS INITIALIZATION
032D  E5        BR50: PUSH H
032E  ED42            SBC HL,BC
0330  E1              POP H
0331  2807            JRZ,BR51        TEST END OF PROGRAM
0333  0A              LD A,(BC)       RECORD BYTE
0334  CD4A03          CALL TRANSM
0337  03              INC BC
0338  18F3            JR BR50
033A  7B        BR51: LD A,E          RECORD CHECKSUM
033B  CD4A03          CALL TRANSM
033E  0680            LD B,80
0340  3E00      BR52: LD A,00
0342  CD4A03          CALL TRANSM     RECORD ZEROES FOR ABOUT 5 SEC.
0345  10F9            DJNZ,BR52
0347  C33E00          JP EXM
```

** TRANSM **

TESTS BITS TO BE RECORDED, GENERATES PROTOCOL : O-BYTE-O

```
034A  C5              PUSH B
034B  0609            LD B,09          BIT POINTER INITIALIZATION
034D  1600     BR53:  LD D,00          BIT TO BE RECORDED = 0
034F  CD6803   BR54:  CALL GENFREQ     RECORD START BIT AND ALL SUCCESSIVE BITS
0352  1007            DJNZ,BR55
0354  1600            LD D,00
0356  CD6803          CALL GENFREQ     RECORD STOP BIT
0359  C1              POP B
035A  C9              RET              RETURN
035B  CB7F     BR55:  BIT 7,A          TEST BIT = "0" OR "1"
035D  2806            JRZ,BR55
035F  1C              INC E            INCREMENT CHECKSUM
0360  1601            LD D,01          BIT TO BE RECORDED = 1
0362  07              RLC A            NEXT BIT
0363  18EA            JR BR54          JUMP TO RECORD "1"
0365  07       BR56:  RLC A
0366  18E5            JR BR53          JUMP TO RECORD"0"
```

** GENFREQ **

GENERATE 1200 HZ FREQUENCIES FOR "0" AND 2400 HZ FOR "1"

```
0368  F5   GENFREQ:  PUSH A           SAVE REGISTERS
0369  C5             PUSH B
036A  CB42           BIT 0,D          TEST BIT TO BE RECORDED
036C  2804           JRZ,BR57         JUMP TO BIT = "0"
036E  3EAA           LD A,AA          BIT = "1"
0370  1802           JR BR58
0372  3E99    BR57:  LD A,99
0374  0610    BR58:  LD B,10
0376  D348    BR59:  OUT(48),A        RECORDING
0378  0E18           LD C, 18         DELAY ½ PERIOD (AT 2400HZ)
037A  0D      BR60:  DEC C
037B  20FD           JRNZ,BR60
037D  07             RLC A            NEXT ½ PERIOD
037E  10F6           DJNZ,BR59        JUMP TO RECORD
0380  C1             POP B            RESTORE REGISTERS
0381  F1             POP A
0382  C9             RET
```

** CARE **

CASSETTE READ

```
0383  1E00     CARE:  LD E,00          CHECKSUM INITIALIZATION
0385  010010          LD BC,1000       START ADDRESS INITIALIZATION
0388  1601            LD D,01          TEST REGISTER INITIALIZATION
```

```
038A  CDAA03           CALL LECFOR    READ LAST LOCATION IN PROGRAM
038D  67               LD H,A
038E  CDAA03           CALL LECFOR
0391  6F               LD L,A
0392  23               INC HL
0393  CDAA03   BR61:   CALL LECFOR    PROGRAM READ
0396  02               LD(BC),A
0397  03               INC BC
0398  E5               PUSH H
0399  ED42             SBC HL,BC
039B  E1               POP H
039C  20F5             JRNZ,BR61      TEST END OF PROGRAM
039E  CDAA03           CALL LECFOR    CHECKSUM READ
03A1  BB               CP A,E
03A2  2003             JRNZ,BR62      TEST ERROR
03A4  C33E00           JP EXM         CORRECT READING
03A7  C3D001   BR62:   JP EXR         ERROR
```

** LECFOR **

BYTE READ

```
03AA  C5     LECFOR:  PUSH B
03AB  0609            LD B,09         BIT POINTER INITIALIZATION
03AD  0E00            LD C,00         BYTE INITIALIZATION
03AF  DB44   BR63:    IN A,(44)       READ
03B1  CB6F            BIT 5,A         SYNCHRONIZATION TEST
03B3  28FA            JRZ,BR63
03B5  CD3301          CALL DEL        DELAY
03B8  CB67            BIT 4,A         TEST Bit = 1 OR 0
03BA  280D            JRZ BR65
03BC  1C              INC E           INCREMENT CHECKSUM REGISTER
03BD  CB42            BIT 0,D
03BF  2804            JRZ,BR64
03C1  CB3A            SRL D           TEST REGISTER = 0
03C3  0605            LD B,05         BIT POINTER REINITIALIZATION
03C5  3E01   BR64:    LD A,01
03C7  1806            JR BR66
03C9  CB42   BR65:    BIT 0,D
03CB  20E2            JRNZ,BR63       LOOP
03CD  3E00            LD A,00
03CF  B1     BR66:    OR C            DECODE BYTE
03D0  1005            DJNZ,BR67       TEST END OF DECODING
03D2  CD3301          CALL DEL        STOP BIT DELAY
03D5  C1              POP B
03D6  C9              RET             RETURN
03C7  CB27   BR67:    SLA A           NEXT BIT
03D9  4F              LD C,A
03DA  18D3            JR BR63         LOOP
```

LOOK UP TABLE

TRANSCODING

03E0	03
03E1	07
03E2	0B
03E3	0F
03E4	02
03E5	06
03E6	0A
03E7	0E
03E8	01
03E9	05
03EA	09
03EB	0D
03EC	00
03ED	04
03EE	08
03EF	0C

LOOK UP TABLE

SEGMENT CODES

03F0	3F
03F1	06
03F2	5B
03F3	4F
03F4	66
03F5	6D
03F6	7D
03F7	07
03F8	7F
03F9	67
03FA	77
03FB	7C
03FC	39
03FD	5E
03FE	79
03FF	71

ANNEX III

RST "RESTART" INSTRUCTIONS

The Z-80 has eight RST instruction addresses and a NMI vector address which is used in the PRO-80 for single step purpose. Upon execution of the RST instructions, the processor saves the PC content on the stack pointer and generates a jump to addresses located in the monitor memory area. Other jumps to RAM addresses have been provided allowing the user to include a second jump to a service routine located anywhere in the RAM. RST instruction cross-references have been summarized in the following table.

INSTRUCTION	MONITOR ADDRESS	RAM ADDRESS
RST0	0000H	-----
RST8	0008H	13BBH
RST16	00010H	13BEH
RST24	0018H	13C1H
RST32	0020H	13C4H
RST40	0028H	13C7H
RST48	0030H	13CAH
RST56	0038H	13CDH

LIMITED WARRANTY

All PROTEC Systems assembled and tested at our facility have a 90 day limited warranty for parts and labour provided the defect is not due to misuse of product.

PROTEC Kits also have a 90 day warranty for PARTS ONLY but are subject to a $40.00 minimum service charge for labour. PROTEC will advise you in writing if the service charge exceeds this amount.

This limited warranty becomes EFFECTIVE upon receipt by PROTEC of the following WARRANTY form. Fill it out and include the name of your PROTEC Product Distributor.

WARRANTY

NAME:..

ADDRESS:.....................................

MODEL:.......................................

SERIAL NUMBER:...............................

DATE OF PURCHASE:............................

DISTRIBUTOR:.................................