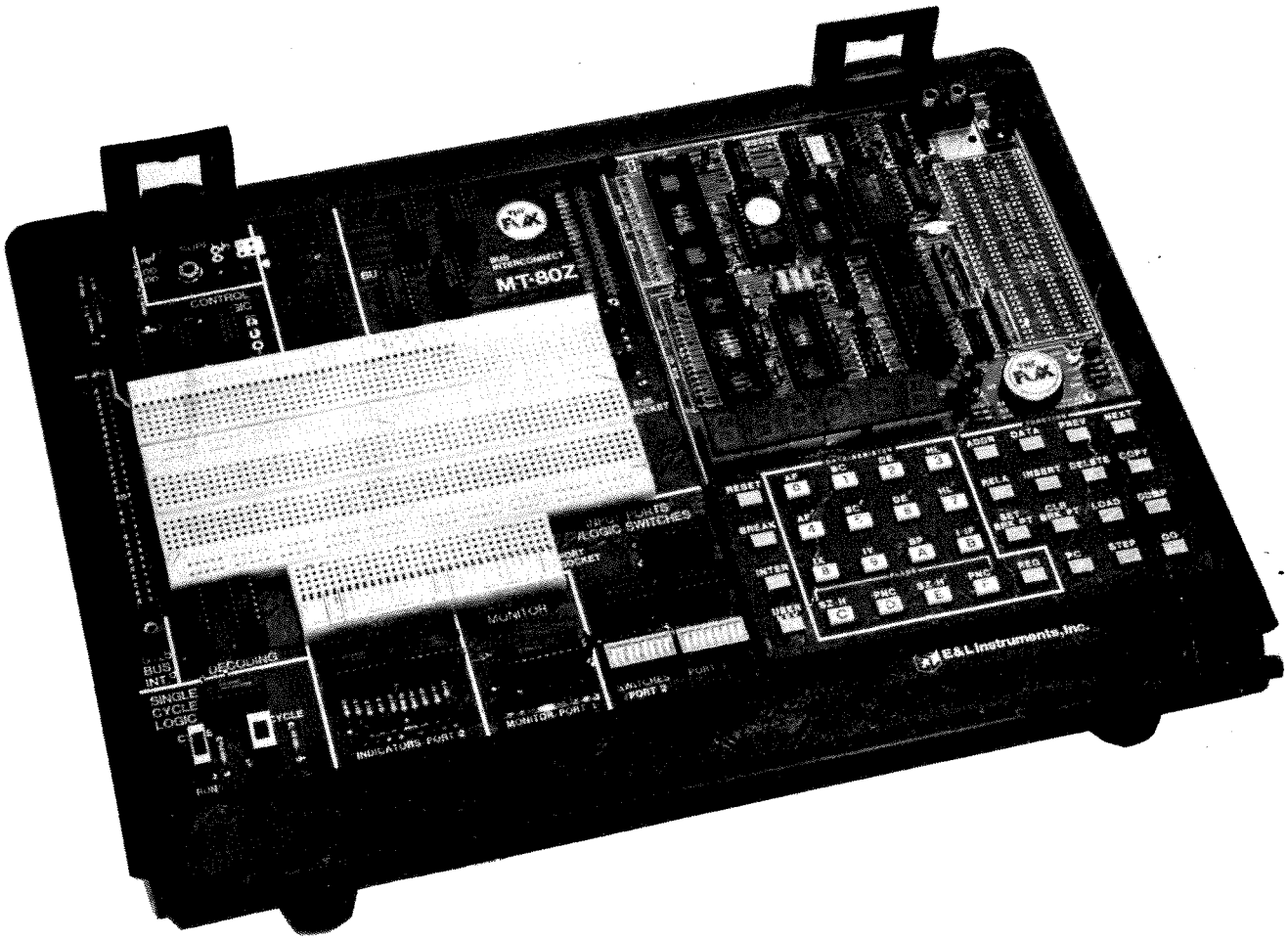
MT-80Z User Manual



801-0246
REV. -
8/82

E&L Instruments, Incorporated
61 First Street, Derby, Connecticut 06418

WARNING

****

FEDERAL REGULATION (PART 15 OF FCC RULES) PROHIBITS THE USE OF COMPUTING EQUIPMENT WHICH CREATES RADIO OR TV INTERFERENCE

****

E & L Instruments specifically warns the user of this instrument that it is intended for use in a classroom or laboratory environment for the purpose of learning and experimentation. When building experimental circuits, it may emit interference that will effect radio and television reception and the user may be required to stop operation until the interference problem is corrected. Home use of this equipment is discouraged since the likelihood of interference is increased by the close proximity of neighbors.

Corrective measures:

Interference can be reduced by the following practices.

1) Install a commercially built RFI power filter in the power line at the point where the cord enters the unit

2) Avoid long wires. They act as antennas

3) If long wires must be used, use shielded cables or twisted pairs which are properly grounded and terminated

807-0018

P R E F A C E

We are in the midst of a microcomputer revolution. Yesterday's science fiction is today's scientific fact. The unique aspect of this technological explosion is that the majority of the developments have occurred within our lifetimes and there is no sign of a slowdown.

Current efforts in microelectronics have created microcomputers or microprocessors on a single silicon chip containing approximately 500,000 transistors. These chips are used in increasing numbers of applications. Each new application puts increasing demands on technical employees. Engineers and technicians in all fields, experimental psychologists, computer programmers, educators, and physicians, to name just a few, feel the need to expand their technical horizons to include a working knowledge of the microcomputer.

The MT-80Z microcomputer described in this User Manual is a complete computer designed to be used as a basic tool for both education and product design. Unlike some computers the MT-80Z is a computer turned inside-out exposing all the parts to provide easy access and facilitate understanding of each section. Once the user has gained substantial knowledge of MT-80Z operation, the computer can be used very effectively for learning computer interfacing, system design and programming.

The MT-80Z is based on the Z80* microprocessor. This chip was introduced by Zilog in 1976. Since that time, the Z80 has become part of a family of Zilog products; namely, the Z8 single chip microcomputer and the 16-bit Z8000 microprocessor. Skills gained using the Z80 based MT-80Z will be easily applied to the Z8 and Z8000. In addition, Z80 users also find it very easy to use microprocessors such as the Intel 8080, 8085, Motorola 6800 and Mostek 6502.

This manual is the work of several people. Chapters 1-3 and Appendix 2 were written by Larry Ryan. The Technical Appendices were put together by Matt Veslocki and draw heavily on material supplied courtesy of Multitech International, and Zilog Corporation. Tom Lingdell drew the schematics and the manual as a whole was edited by Andrew Singer.

*Z80™ is a trademark of the Zilog Corporation.

TABLE OF CONTENTS

TABLE OF CONTENTS

CHAPTER 3 (Continued)

A P P E N D I C E S

APPENDIX 1

Hexadecimal, Decimal, Binary Number Chart

APPENDIX 2

MT-80Z Keyboard Quick Reference List

APPENDIX 3

Z-80 Reference Manual

APPENDIX 4

MT-80Z Specifications

APPENDIX 5

MT-80Z Schematics

APPENDIX 6

MT-80Z Memory Expansion

APPENDIX 7

MT-80Z Keyboard and Display

APPENDIX 8

MT-80Z Monitor Program Source Listing

APPENDIX 9

MT-80Z Parts List

Effective study requires all the senses so GO SLOW! Try to perform each step and note all results. After going through Chapters 1-3 carefully you should be able to do the following:

1. Given a Z80 machine language program, enter, single step and edit the program and interpret the results.

2. Identify, locate, and describe the functions of the major blocks of the computer.

3. Utilize the PORT and BUS sockets for input and output interfacing.

4. Given address and number of bytes, save and load memory contents on an audio tape recorder.

5. Given an improperly written program, locate program faults using break-points, single instruction stepping, single cycle stepping and the bus monitor display.

6. Using the appropriate keys, observe and/or change Z80 register contents or flags.

The MT-80Z User Manual is concerned mainly with MT-80Z use and operation. It does not provide a comprehensive tutorial on microcomputer technology. If you are just starting out, and you find that even Chapters 1-3 present difficulties, or if you're interested in digging deeper into this subject, you should consider going through E & L's "Introduction to Fundamentals of Microcomputer Programming and Interfacing", Modules 1, 2A, and 2B of the FOXWARE SERIES, E & L Part number 345-8001(Covers the entire set). If you are new to microcomputers, you will find the MT-80Z and the FOXWARE tutorials to be a superior learning tool for your entry into the world of microcomputers. If you are a seasoned veteran in this field, you will find the MT-80Z to be an excellent system for product design, program design or breadboarding.

How To Use This Manual

This manual was created to provide you with all the information you need to make effective use of your MT-80Z Microcomputer Trainer. The information in the manual is presented in several ways so that, depending on your background and level of technical expertise, you can choose the approach that suits you best.

If you are just starting out with micros, you should go through Chapters 1 to 3 in that order. The first chapter is a description of the major sections of the computer. If you have the MT-80Z in front of you now, notice that each section is marked off and labeled for convenient recognition. The second chapter is designed to allow you to operate some of the basic features of the MT-80Z right away. This gives you a quick orientation to microcomputer operation and provides an easy procedure for checking the operation of those features of the computer which only require keyboard access. This chapter is written in a detailed "by-the-numbers" fashion. Finally, Chapter 3 consists of a series of experiments designed to give you experience in the operation of every control and socket except for the STD BUS socket. All three chapter are intended for a reader without much technical background. You may find Appendices 1,2 and 3 helpful in conjunction with the chapters.

If you are fairly familiar with micros, but not really an expert you should probably go through Chapters 1 and 2, and then play with the keyboard functions using Appendix 2 as a reference. You will find the other Technical Appendices helpful for specific information of a more detailed nature.

If you are expert with micros, start out by looking at the Specifications(Appendix 4), Schematics(Appendix 5), Keyboard Quick Reference List(Appendix 2), and perhaps the Monitor Source Listing(Appendix 8). You may wish to look at Chapter 1 for an overview of the unit in written rather than specification form.

When going through the material in Chapters 1-3, you will find that the steps you are asked to perform are presented in highly detailed form. To make things clear, most of the questions are answered. You could easily rush through the material, skipping steps and consequently learning very little.

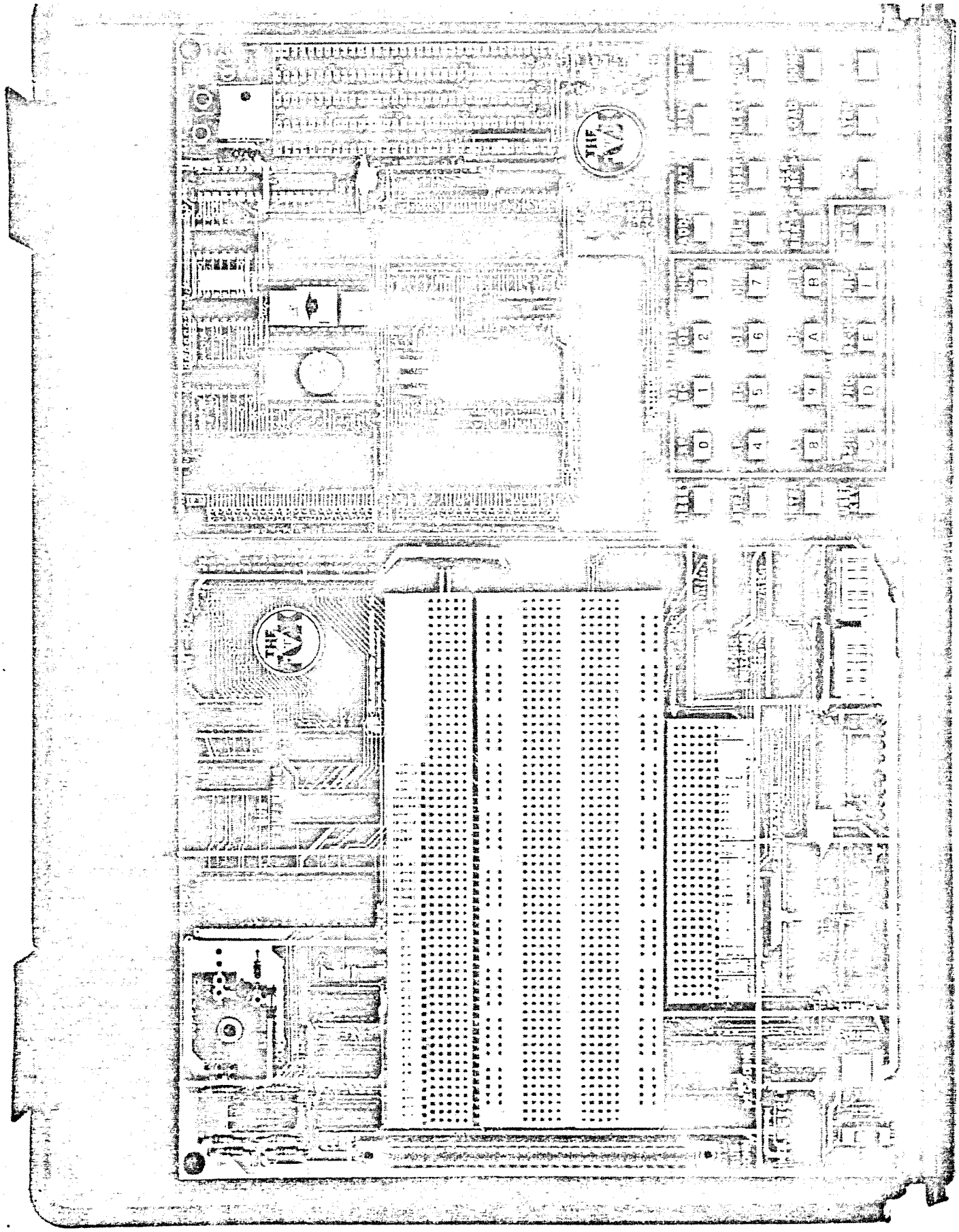


Figure 1-1. Top View of MT-80Z

CHAPTER 1

THE MAJOR SECTIONS OF THE MT-80Z

Introduction

The purpose of this chapter is to provide a description of each major section of the MT-80Z Microcomputer Trainer. All the sections described are visible and clearly marked. There are no hidden components to prevent you from eventually understanding how the computer operates. As each section is described, it is advised that you locate it on your computer. The term, computer architecture, is often used to describe the method in which the various sections of a computer system are configured and interconnected. The basic architecture of all computers consists of sections for CPU(Central Processing Unit), Control, Memory, and Input/Output. The CPU and Control sections are the "brains" of the system, performing arithmetic and logical processes and regulating the flow of data within the computer. Memory provides storage for programs and data used by the CPU. The Input/Output or I/O section provides the means to communicate with the computer. When you use the keyboard to enter a program or generate tones from the MT-80Z speaker, the I/O sections of the computer are being activated.

The description of MT-80Z architecture that follows will be more detailed than the general computer requirements described above. You will be taken on a guided tour of all the sections labeled in white lettering on the top of your micr computer. If you have jumped ahead a little and already applied power, turn it off now so that you can sit back, hold the MT-80Z in a comfortable position for easy viewing, and read this manual.

Objectives

After completing this chapter you will be able to:

1. Demonstrate your understanding of MT-80Z architecture by describing each of the major sections of the computer and how they relate to each other.

2. Interpret the results displayed on the Port 1 and 2 display LEDs.

3. Interpret the 7-segment numeric display LEDs.

4. Locate and identify the function of the various interfacing sockets.

5. Understand the basic functions available on the keypad.

6. Describe the methods of applying power to the MT-80Z.

7. Understand the use of the Port 1 and 2 logic switches.

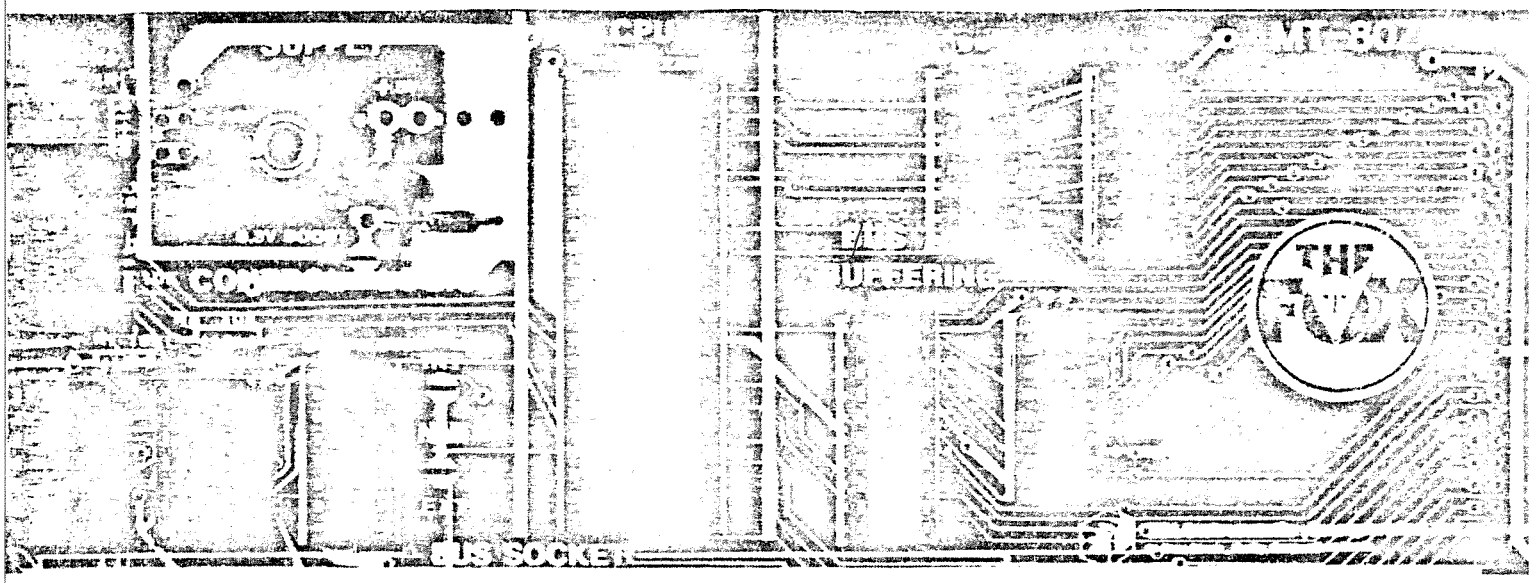


Figure 1-2. Top Left Hand Corner of MT-80Z Panel

CPU

This section consists of a single 40-pin chip: the Z80 microprocessor. As in any microcomputer, the microprocessor determines the power and capabilities of the computer system. The microprocessor responds to instructions and data stored in memory, performs the required operations and provides the necessary control signals. The speed of the Z80 is determined by the clock crystal located on the right-hand section of the computer. The crystal is marked 3.579 MHz. The Z80 divides the crystal frequency by 2 giving a CPU cycle time of approximately 559 nanoseconds. This allows the MT-80Z to add two binary numbers in little more than 2.2 millionths of a second! For writing MT-80Z programs the Z80 instruction set is the primary tool.

Control Logic

The control logic section is located in the top left corner of the MT-80Z. This section provides buffering (current amplification) for the control signals and logically derives some additional control signals not generated by the Z80. The collection of control signal connections is called the control bus. The term bus indicates that these signals are to operate in "sync" with the computer. The buffering allows the control signals to drive a large number of logic inputs for circuits that are interfaced with the Z80. The additional control signals (MCSYNC, and INTAK, for example) are required by the STD bus which will be discussed later.

An additional function located in the Control Logic area deals with user options in the use of NMI(Non-Maskable Interrupt). Interrupts are a scheme whereby an asynchronous (not in sync with the Z80 clock) input may interrupt the Z80 and cause it to immediately run another program. After the interrupt program is finished, the Z80 can be easily returned back to the main program.

The NMI is the Z80's highest priority interrupt and can be used in the MT-80Z in two different ways. First of all, look closely in the Control Logic area and find the letters NMI. Notice also the two solder pads labeled INT and EXT. The INT or Internal pad should be already soldered. The EXT or External pad should be clean. When the INT pad is soldered, the Z80 gets an NMI from the BREAK key on the keypad. If EXT is soldered instead, NMI is to be applied by a circuit plugged into the STD bus socket. It is advised that only one option be used at a time.

+/- Supply

The +/- Supply section is located near the CPU and Control Logic section. The purpose of this section is to provide an auxiliary ± voltage power supply. It is important to note that the wall-mount A.C. adapter supplied with the MT-80Z DOES NOT plug into this area. It plugs into the POWER jack on the right-hand side. There is really no danger of plugging in the wrong supply because different connectors are used. The voltages brought into the MT-80Z are connected to regulator I.C.s and are used to power the computer. The regulated voltages are also available on the BUS and PORT sockets for interfacing experiments.

Bus Buffering

In addition to the control bus, the microcomputer uses data and address buses. The address bus consists of 16 connections which are used to transfer information to the memory or I/O elements. This bus is a one-way street or unidirectional and can be used to communicate 2^{16} or 65,536 possible memory addresses. Each memory or I/O element is identified by one of these addresses. When the Z80 wants to fetch information from memory, it broadcasts the address on the address bus, and the memory chips send back an 8-bit binary number on the data bus.

The data bus is a 2-way street or bidirectional bus allowing 8-bit data to flow into or out of the Z80. The Z80 indicates the direction via the control bus.

MATRIX CHART

STD CARD FUNCTIONS AVAILABLE

- ◉ AVAILABLE FUNCTIONS
- ● PLANNED FUNCTIONS
- — SOME CARDS DO MORE THAN ONE FUNCTION
- — SOME MANUFACTURERS HAVE MULTIPLE VERSIONS OF VARIOUS FUNCTIONS

		ANALOG DEVICES, INC	ANTONA CORP.	APPLIED CONTROL TECHNOLOGY	APPLIED MICRO TECHNOLOGY	ATEC	ATEC SYSTEMS	AUCAT	BARDINE PRODUCTS	CONCORDS AND SYSTEMS	CONTEMPORARY CONTROL SYSTEMS	DATA TRANSLATION	DATRICON CORP	DESIRT MICROSYSTEMS	DIGITAL DYNAMICS, INC	DIMAS INC	DOUGLAS ELECTRONICS	E&L INSTRUMENTS INC	ELECTROLOGIC	ENERGY SECURITY SYSTEMS	ENLOOL INCORPORATED	INTERAVISE SYSTEMS CORP	FORETHOUGHT PRODUCTS	HARTMANN - LANG	I/O CONTROLS INC	INTELLIGENCE SYSTEMS LTD	INTERMAGNETICS GENERAL	INTERSIL INC	INTRA COMPUTER INC	KENNOLY COMPANY	MATRIX CORP	MATROX ELECTRONIC SYS.	MICROCOMPUTER SYSTEMS	MICRO LINK CORP	MICRO SYS INC	MOSTEK	PROLOG CORPORATION	SAMCO	SBTRONR SYSTEMS INC	SPINNER PERIPHERALS	TANDEM CONTROLS	TECHNITRON (UK)	TECHNONICS	TEXAS INSTRUMENTS	VECTOR ELECTRONIC CO	WHELCO INC	XITEX	ZIATECH CORP	ZYDYCO INC
CPU	Z80				●											●												●								●	●				●								
	8085																							●				◉									●											●
	6800												●																		●						●												
	6502												●									●	●															●											
	6809												●									●									●																		
	8088													◉											●								◉															◉	
	6802																					●									◉																		
	8080A										◉																						◉																
	NSC 800								●																																								
MEMORY FUNCTIONS	STATIC RAM				◉						◉					◉						●						◉								●	●											
	DYNAMIC RAM				◉									●										●				◉								●													
	BATTERY BACKUP RAM														◉	◉						●									◉			●		●	●											●	◉
	EEPROM																																																
	EPROM				◉						◉			●	◉	◉						●						●						●		●	◉											◉
	BUBBLE MEMORY																																									●	◉					
DIGITAL I/O FUNCTIONS	TTL INPUT		◉		◉															●								◉									●											
	TTL OUTPUT		◉		◉															●								◉									◉											
	TTL I/O (PIO)		◉							◉	◉		◉										●					●			◉		●			●	◉												◉
	LINE DRIVER/RCVR		◉		◉												●																		◉													
	SWITCH INPUTS					◉				◉																																				◉			
INDUSTRIAL I/O FUNCTIONS	RELAY OUTPUT						●													●								◉								◉											◉	●
	DRIVER OUTPUT																																				◉												
	AC/DC OPTO-INPUT				◉											◉				●							◉	◉			●			◉		●	◉		◉										
	AC OPTO-OUTPUT															◉				●								◉			●			◉		●	◉		◉										
	IEEE 488				●																							●			●																◉	
	FIBER OPTIC (SI/O)																										◉																					
	MODULE I/O																		●																		◉											
	RS-422		●																																													●
ANALOG I/O FUNCTIONS	A-D INPUT				◉					●		◉								●								◉						◉		◉												
	D-A OUTPUT				◉							◉								●								◉								◉												
	A-D/D-A SYSTEM	◉								◉																		◉								◉												
	MUX & EXPANDER						●					◉																																				
	SERVO CONTROLLER			◉																																												
	TEMPERATURE INPUT																																								◉							
	SIGNAL CONDITION					◉																																										
PHERIPHERAL INTERFACE CONTROLLER FUNCTIONS	UART (SIO)		◉		◉						◉									●					◉		◉	◉	◉		◉			◉	◉	◉	◉	●								●		◉
	CRT		◉		◉																	●	●					◉				◉		◉						◉							●		
	DISK		●		◉																		●					◉		◉	●			●	◉	◉				◉							●		
	TAPE																				●				◉					◉				◉									◉				●		
	PRINTER		◉																					◉				●						◉													●		◉
	KEYBOARD		◉								◉										◉	●												◉			◉										●		
	DISPLAY					◉					◉										◉				◉						◉	◉		◉			◉												
	MODEM								●							◉						●												◉															
	PROM WRITER		●		◉								●															◉						◉						◉									
	SDLC																																		◉													
	OTHER BUS INTERFACES																						◉	◉																					◉			
	STEPPER MOTOR																																													◉			
SPECIAL FUNCTIONS	DMA CONTROLLER													●																					◉													
	MATH				◉									●														◉	●					◉		◉													
	COUNTER		◉		◉	●				◉			◉		◉	◉					◉										◉			◉	◉	◉	◉									◉			
	CLOCK/TIMER																				●										●		●		◉	◉	◉											◉	
	INTERRUPT CONTROL		◉		◉					◉			◉		◉	◉					◉										◉			◉	◉	◉	◉		◉										
	SPEECH SYNTHESIZER															●																																	
	POWER FAIL																				●										●																		
HARDWARE SUPPORT FUNCTIONS	MOTHERBOARDS				◉			◉			◉		◉					●						◉							◉						◉			◉				●					
	EXTENDER							◉	◉				◉				◉											◉			◉					◉	◉			◉				●					
	PROTOTYPE CARDS							◉	◉								◉	◉							◉			◉								◉	◉							◉					
	DIAGNOSTIC CARDS												◉					●	●									●						●		◉													

Figure 1-3. STD Product Chart

The Z80 produces very small drive currents at its pins. The buffering amplifies the current allowing the connection of many devices to the pins and provides protection of the Z80 by isolation.

STD Bus Connector

The STD bus connector, J1, is located on the left hand edge of the MT-80Z. This 56 pin edge-card connector will receive one of the hundreds of pre-built modules built on standard 6.5 by 4.5 inch printed circuit cards which use this popular, industry-standard bus. By inserting the card into J1, it is automatically connected to the MT-80Z data, address, and control buses and with appropriate jumpering can receive +5V and +/-12V power. STD cards can be added to the MT-80Z to enhance its capabilities, provide training on STD subsystems or to design and test STD circuits. There are over 100 different manufacturers producing a wide variety of STD Bus cards. Figure 1-3 shows some of the many types of STD products available and who manufactures them.

Detailed information on STD products and the STD bus standard is available from the IEEE P961 STD Bus Working Group, STDUSR (The STD User's Group), 8697 Frobisher Street, San Diego, CA 92126, or E & L Instruments.

J2 - An Accessory Bus

J2 is located near the top center of the computer. It will appear to you as two rows of holes in the printed circuit board. The purpose of J2 is to allow you to solder an appropriate connector and plug in accessories. Figure 1-4 shows J2. J2 is similar to the STD bus in that the 40 connections will provide access to the address, data, control and power buses.

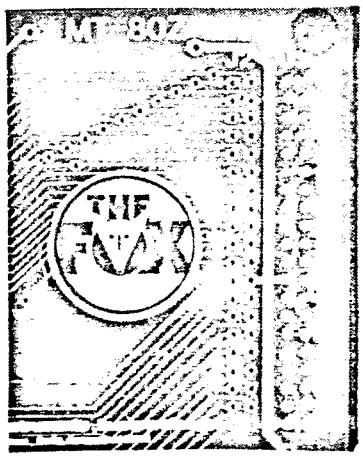


Figure 1-4. J2 Region of MT-80Z Front Panel

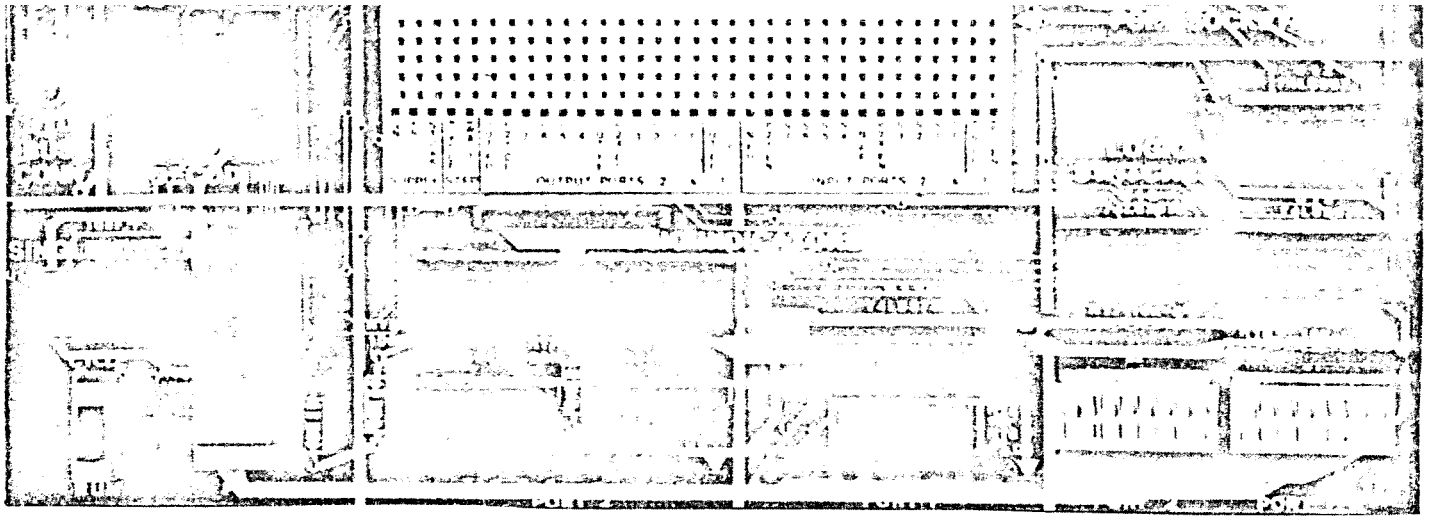


Figure 1-5. Bottom Left Side of MT-80Z Front Panel

Decoding

The decoding section consists of two chips located between the bottom of the STD connector and the Port socket. The specific purpose of this section is to use signals from part of the address and control bus to generate the special sychronizing pulses needed for I/O data transfers. These pulses are made available on the Port socket and their use is explained later.

Single Cycle

The Single Cycle section is located in the bottom left corner of the computer. It contains two switches: one a push-button and the other a slide switch. Through proper use of these switches, you may "step" the Z80 one machine cycle at a time. This slows the computer sufficiently so that you can observe the detailed operations of the Z80 and associated interface hardware.

In addition to regular full speed program execution, there are two methods provided by the MT-80Z which allow you to step through a program. One steps by instruction and the other steps by cycle. A single instruction can consist of several machine cycles. Here is a summary of all three methods to run a program:

1. Regular Full Speed: Press the GO key on the keypad. The program executes at computer's full clock speed.

2. Single Instruction Step: Use the STEP key on the keypad. Each STEP will execute several machine cycles but only one instruction. This method is useful for program debugging.

3. Single Cycle: Switch the CYCLE-RUN switch to CYCLE and push the large push-button (PB1) switch to perform one machine cycle at a time. This stepping method is the slowest of the three and is most useful when you need to observe how your hardware is responding to program execution. NOTE: Single cycle will not function unless WAIT and CYCLE are jumpered on the PORT SOCKET.

Logic Indicators/Output Ports

The Logic Indicators can be found next to the Single Cycle section at the bottom of the computer. There are two rows of 8 LEDs labeled Port 1 and Port 2. Note how they are separated in groups of four LEDs for easy binary to hexadecimal number conversion. When read as a binary number, the least significant bit is to the right and is numbered "0".

The Logic Indicators provide the following functions:

1. Simple logic monitoring.

2. Data bus monitoring during single cycle operation.

3. Latching and displaying bus data sent from the Z80 using the series of OUT instructions.

Simple logic monitoring is done by connecting wires from L0-L7 of the PORT SOCKET to any logic output from the MT-80Z or interface circuits. The results are viewed on the PORT 2 LEDs. The LED marked 0 will correspond to L0 on the PORT SOCKET and LEDs 1-7 correspond to L1-L7.

Data bus monitoring uses the PORT 1 display. These LEDs are permanently wired to the data bus and are not available on the PORT SOCKET. What this means is that for every Z80 machine cycle, the Port 1 LEDs will display the data traveling on the bidirectional data bus. Used in conjunction with Single Cycle operation, the bus monitor can enable you to observe the contents of the data bus each time you push the CYCLE button (PB1). This is the fastest way for you to learn how the Z80 actually works. It is also an excellent microcomputer troubleshooting method.

Port 1 and Port 2 Logic Indicators can be used as output ports. This use of the Logic Indicators allows you to latch and display register or memory data sent to the data bus as a result of an OUT instruction. The format of the OUT instruction requires a port number. The port numbers available are FD, FE, and FF. Using appropriate jumpers on the PORT SOCKET, you can assign any of the three addresses to the ports.

Another feature of Port 2 allows you to split the 8-bit display into two separate 4-bit displays. When split, the halves are designated P2X and P2Y; port 2X and port 2Y. The 4-bit ports can be assigned different port numbers.

Logic Switches/Input Ports

The Logic Switches, located at the right of the Logic Indicators, consist of two DIP switches marked Port 1, S1, and Port 2, S2.

The Logic Switches provide the following functions:

1. Port 2 -- Apply a logic 1 or 0 to PORT SOCKET terminals S0-S7.

2. Port 1 and 2 -- Supply data to the Z80 as input ports.

Digital experimentation often requires a steady-state logic level (1 or 0) applied as an input to a circuit. The Port 2 (S2) logic switches can be used for this purpose. The eight different switches control logic levels at PORT SOCKET terminals S0-S7. These terminals can be connected to the inputs of interfacing circuits.

Both ports 1 and 2 can be configured as input ports to the Z80. Port 1 (S1) is already wired to the data bus. Used as an input port, it requires a jumper between P1 EN (port 1 enable) and IN FF on the port socket. This assigns the port address FF to port 1. An instruction such as IN A,(FF) will input the switch settings as an 8-bit binary number to the Z80 accumulator.

The configuration of port 2 as an input port requires some jumpering between the PORT SOCKET and BUS SOCKET. Port 2 may be addressed FD or FE by applying jumpers on the PORT SOCKET. Another feature of port 2 allows you to split the port into two independent groups of four switches each. When split, port 2 becomes 4-bit port 2X (P2X) and 4-bit port 2Y (P2Y).

Note the numbering on the printed circuit board near the switches: 76543210. These numbers designate the position the switch will occupy on the PORT SOCKET (S2) or data bus (S1). For example, S2 switch 0 corresponds to PORT SOCKET connection S0. The switches themselves may have additional numbering: 87654321. These numbers still correspond to conventional positioning, i.e., least significant bit to the right. To select a logic 1, move the rocker toward "OPEN" and for a logic 0, move it toward the number.

20

Port Socket

The Port Socket is located just above the Logic Indicators. It consists of 33 vertical rows of connectors to give you access to power, LEDs, and switches. It allows the jumpering required to establish addresses for Ports 1 and 2. The socket can be divided into four sections:

1. Supply
2. Step
3. Output Ports 1 & 2
4. Input Ports 1 & 2

These sections correspond to the organization of the socket from left to right. Before you ready any further, examine the label on the socket and shown in Figure 1-6 below.

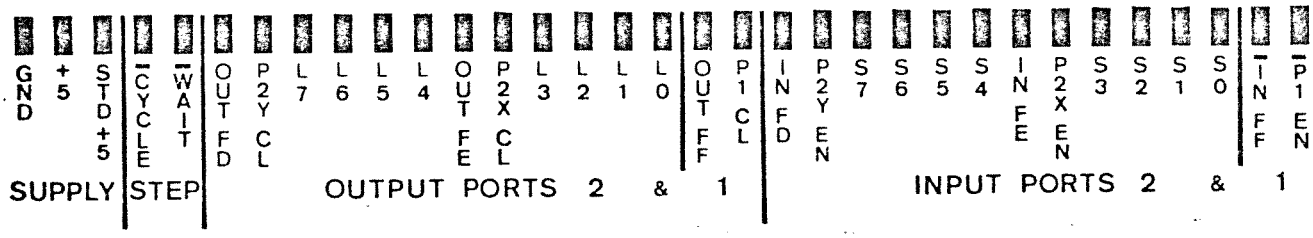


Figure 1-6. Port Socket Label

The Supply section provides access to the MT-80Z internal +5V power supply and ground for powering interface circuits. These circuits could be mounted on the large breadboarding socket (SK-10) mounted above the Port Socket. The connection marked STD +5 is connected directly to pins 1 and 2 of the STD bus edge connector socket. This gives you the option of providing +5V power to the STD bus from the MT-80Z power supply by jumpering to +5V.

The two connections in the Step section are CYCLE and WAIT. They must be jumpered together before the SINGLE CYCLE feature will function. Without the jumper, you could supply a WAIT input to the Z80 from some external circuit.

OUTPUT Ports 1 & 2 provide connections to control the Port 1 and 2 Logic Indicators. Their functions are outlined below:

1. L7-L0: Inputs to the Port 2 LEDs. These inputs are buffered and latched by chips U15 and U16.

2. OUT FD, OUT FE, OUT FF: These connections are outputs from the Decoder section. They are used to establish the addresses of the Port 1 and Port 2 displays. Also, they can be used with other interface circuitry.

21

3. P1 CL: This is the symbol for Port 1 Clock. The term clock is used here to signify an enabling pulse for the Port 1 latch chip. This connection is an input.

4. P2X CL, P2Y CL: These are the symbols for Port 2X Clock and Port 2Y Clock. Remember that Port 2 can be split into sections X and Y. This is allowed by having two separate inputs for the port latches.

Input Ports 1 & 2 are very similar in operation to the output ports described above. The major difference is that switch outputs are provided instead of display inputs. Here are the functions:

1. S7-S0: Outputs from DIP switch S2, Port 2. The voltage levels at these connections will be +5V or ground depending on switch positions. Remember that S1 switches are already interfaced to the data bus.

2. IN FD, IN FE, $\overline{\text{IN FF}}$: These outputs from the Decoder section are used to establish addresses for the Port 1 and 2 switches and other interface circuits. Note that $\overline{\text{IN FF}}$ is active-low and should be used with Port 1.

3. P2X EN, P2Y EN, $\overline{\text{P1 EN}}$: These are the ENABLE inputs for the tristate buffers that interface the switches to the data bus or S7-S0. Note that one of them, PORT 1 ENABLE, is active-low. It will be most convenient to "map" this port at address FF using the active-low $\overline{\text{IN FF}}$.

Bus Socket

The Bus Socket is the long, narrow white socket located above the SK-10 breadboarding socket. All but two of the connections shown on the label are wired directly to the STD BUS edge connector. The +12V and -12V connections are used to provide these voltages to the STD bus AUX +V and AUX -V. The 12V source comes from the Supply section when the optional +/ - power adapter is used. Jumpers are used to make the appropriate connections.

There are three main groups of connections on the Bus Socket. Access to these bus groups is necessary to interface other components to the MT-80Z.

1. A15-A0: The computer address bus.
2. D7-D0: The computer data bus.
3. $\overline{\text{SY RST}}$ to REFRESH: The computer control bus.

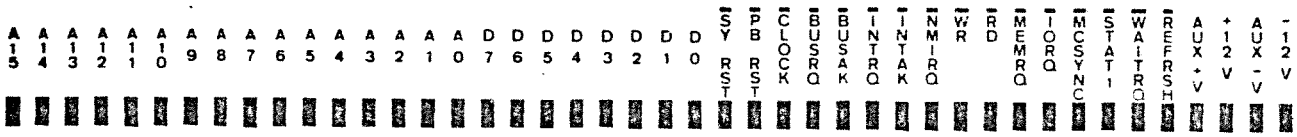


Figure 1-7. Bus Socket Label

Tape Recorder Interface - EAR, MIC

There are two miniature phone jacks located at the top right corner of the MT-80Z. These jacks, marked EAR and MIC are connected to a cassette tape recorder so that you may save and load programs. By using the LOAD and DUMP keys on the keypad, the binary information stored in memory is converted to tones that may be recorded or played back. This feature is a time saver that can prevent you from having to reload programs using the keypad. Most programs load in a short time. In fact, an entire 2K of memory can be saved or loaded in 2 MIN and 14 SEC.

To make the connection to your tape recorder, plug in cables from EAR on the MT-80Z to the earphone output on the recorder and from MIC on the MT-80Z to the microphone input on the recorder. Cables suitable for this purpose require miniature phone plugs at one end, and plugs suitable for connecting to your cassette recorder at the other end. Typically, such cables are readily available at Radio Shack or hi-fi equipment stores. An experiment in Chapter 2 provides detailed experience with this feature.

Power Connector

Next to the tape recorder jacks, there is a special connector for +5V power. This is where you plug in the A.C. wall mount adapter supplied with the MT-80Z. Be sure to use only the supplied adapter as it has been specially designed for the heavy duty requirements of the computer. The MT-80Z has no ON-OFF power switch. As soon as you plug in the adapter the computer is powered up and running.

Speaker and TONE LED

The small loudspeaker and green TONE LED are located to the right of the 7-segment displays. Both of these are interfaced to the Z80 bus through use of the 8255 Programmable Peripheral Interface chip. This is the large, 40-pin chip near the TONE LED. To make the speaker work, a program is written that uses time delays to form an audio frequency square wave. By adjustment of the program, various tone frequencies can be produced. The TONE LED is connected so that it operates along with the speaker.

The speaker "beeps" each time you press one of the keys on the keypad. This annunciator tone provides some aural feedback to indicate that the key press actually made contact. In the experiments you will learn how to change the "beep" duration and frequency or even eliminate it.

When you use the audio tape feature, the MT-80Z speaker makes the data recording or playback audible. This makes it possible for you to confirm that cassette data is playing back at an appropriate level for the computer.

HALT LED

The red HALT LED located just below the TONE LED indicates when the Z80 is in a HALT state. After executing a HALT instruction, the Z80 outputs a HALT signal which lights the LED. When halted the Z80 stops fetching and executing further instructions.

The visual indication provided by the HALT LED is useful when you need to know when the HALT in your program has been executed. Your programs could have a "bug" that causes the computer to appear hung up and unresponsive. By placing a HALT instruction at strategic locations in your program, you can find out how much of the program executes normally.

Memory

The MT-80Z uses a combination of ROM(Read Only Memory) and read/write memory referred to as RAM. RAM (Random Access Memory) is volatile. This means that when power is turned off, the stored information is lost. When you use the keys to store a program or data, the information is being stored in RAM. The ROM retains its memory regardless of whether the power is on or off.

The memory chips used in the MT-80Z are marked U6, U7 and U8. Socket U7 may be empty. It is there if you want to expand the memory capacity. U6 is an EPROM (Eraseable-Programmable Read Only Memory) that permanently stores all the operating software (programs) that the MT-80Z requires. This software is

generally called the Monitor Program or System Monitor. This rather large program (2K) has as its major task, interpreting key depressions and providing appropriate data to the 7-segment displays. The Monitor contains many small subroutines that you can call from the programs you write. For example, if you would like the speaker to beep at certain points in your program or set up a special display, the Monitor subroutines can make your programming job much easier.

The details of memory expansion and the Monitor program listing are beyond the scope of this Chapter. Please consult the Technical Appendices 6(Memory Expansion) and 8(Monitor Source Listing) for this information. The MEM-80, Full Memory Expansion Package, E & L Instruments P/N 200-8020 is available from E &L for expanding the memory of your MT-80Z.

The technique used to help you gain a mental picture of computer memory allocation is called a Memory Map. The map in Figure 1-8 shows how the MT-80Z memory is allocated. The programs you write and store in memory are located some-where between 1800 and 1FFF. AN IMPORTANT NOTE: Even though the Monitor is located in EPROM (0000-07FF), it still requires some RAM for stack and "scratch pad" use. You should avoid using the area shown on the map starting at 1F9F and ending at 1FF3. The Monitor automatically sets the user stack pointer at 1F9F. Any time you use PUSH, POP, CALL, RET, and interrupts, the stack memory area is affected.

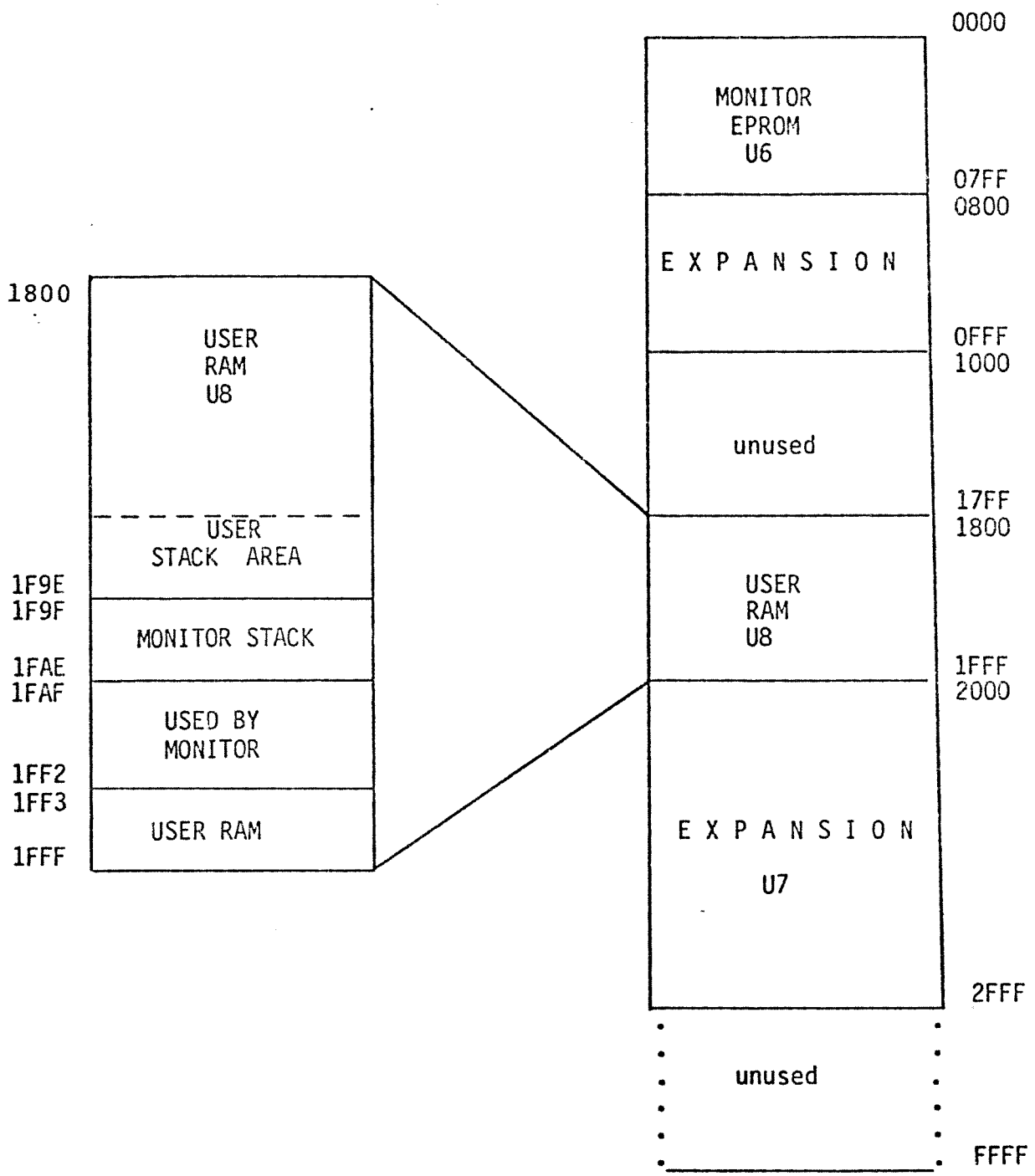


Figure 1-8. MT-80Z Memory Map

8255 PPI

The 8255 PPI(Programmable Peripheral Interface) is U14, the large 40-pin chip located above the 7-segment display. This parallel I/O chip can provide three separate 8-bit I/O ports, 2 ports with "handshaking" or a single 8-bit bidirectional port. The various modes are programmable, making this one of the more versatile I/O chips currently available.

The 8255 is committed to the task of keyboard input and display output. Other uses are the Speaker and TONE LED, Tape Recorder I/O, and the USER key on the keypad. The Chapter 3 experiment on the USER key gives you the opportunity to write and test the software required to operate the PPI.

If you wish to expand the I/O capabilities of the MT-80Z, you can add the Zilog PIO (Parallel Input/Output) and CTC (Counter/Timer Circuit) chips. A description of the 8255 can be found in Technical Appendix 7. I/O expansion details can be found in the MIO-80, CTC/PIO Expansion Package for the MT-80Z, E & L Instruments P/N 200-8030.

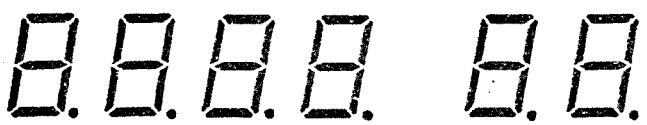
As it does for memory organization, mapping helps to give a clear picture of how the port numbers (I/O addresses) are allocated. Figure 1-9 shows a map of the ports. Remember that the Port 1 and 2 Logic Indicators and Switches are mapped by the jumpers inserted into the Port Socket. The 8255 PPI, PIO and CTC are already wired to fixed I/O addresses. In the programs that you write, try to avoid using these fixed addresses.

	Port Number	
PPI PORT A	00	
PPI PORT B	01	
PPI PORT C	02	
PPI CONTROL	03	
UNUSED		
CTC0	40	
CTC1	41	
CTC2	42	
CTC3	43	
UNUSED		
PIO A DATA	80	
PIO B DATA	81	
PIO A CONTROL	82	
PIO B CONTROL	83	
UNUSED		
USER	FD	PORT SOCKET
USER	FE	
USER	FF	

Figure 1-9. MT-80Z Port Map

Address/Data 7-Segment LED Display

The Address/Data display consists of six 7-segment displays covered with a red filter. They are located just above the keypad for easy viewing. In addition to memory addresses and contents, the display provides a prompting message, a sign-on message and a warning message. The displays are grouped into a 4-character address field and 2-character data field as shown below:



ADDRESS FIELD DATA FIELD

Figure 1-10. 7-Segment Display Format

Numbers are displayed in hexadecimal form. This is a compact method of representing a binary number which is easy to convert. A conversion table can be found in Appendix 1. The MT-80Z address consists of 16 bits which require 4 hex digits: 0000-FFFF. The data bus is 8-bits (1 byte) and requires two digits: 00-FF. The use of address/data and other displays will be explained in the experiments.

The 7-segment display is designed to show the digits 0-9. By careful use of the segments (and a little imagination), many other characters can be represented. A diagram on the bottom of Page 4 in Appendix 7 lists some of the characters the MT-80Z can display in 7-segment form. Figure 1-11 shows some of the more common characters you will use.

The displays are controlled by the Monitor program. Each segment is individually software controllable so that you may write a program that can display any combination of segments and decimal points. Technical Appendix 7 provides detailed information on how the PIA is used to light various segments.

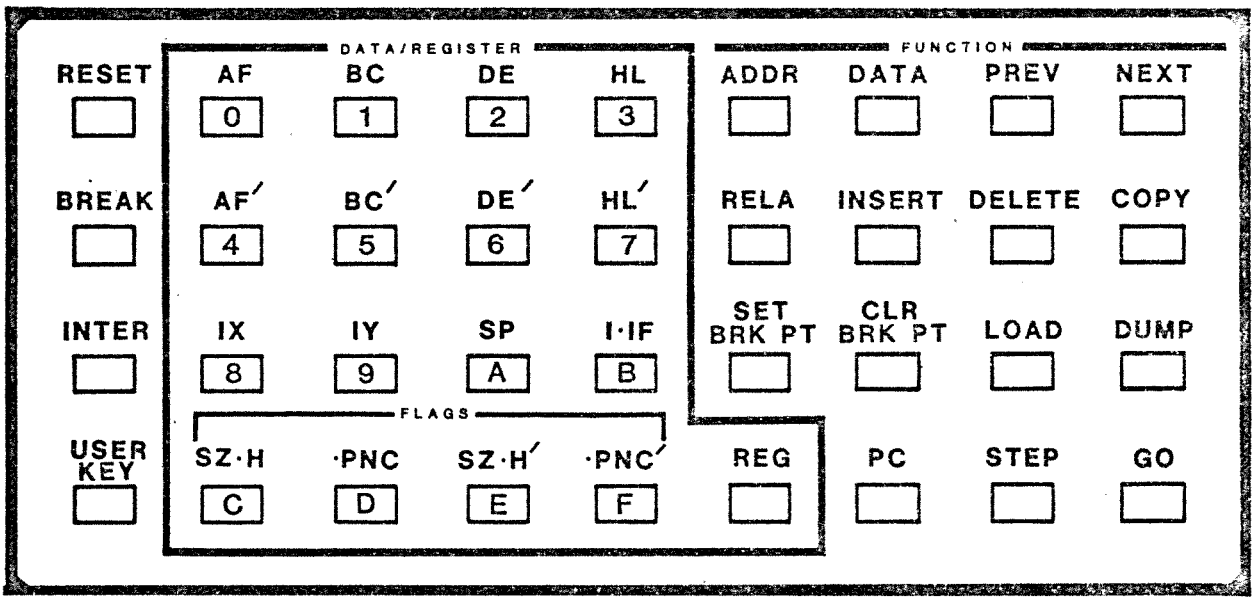


Figure 1-12. MT-80Z Keypad

Keypad

The keypad consists of 36 pushbutton keys arranged in logical groups. The general categories of keypad functions are listed below:

1. Input programs
2. Display memory and all Z80 register contents
3. Preset any Z80 register
4. Run programs
5. Set and clear program breakpoints
6. Edit memory contents
7. Move blocks of data from one part of memory to another
8. Single step by instruction
9. Calculate relative addresses for the Z80 instructions JR and DJNZ
10. Load and store memory contents on tape
11. Halt program execution
12. Interrupt the Z80
13. Utilize a user-definable key
14. Reset the computer

This User Manual provides information about the keypad on four levels of increasing detail. The first and most condensed level is the Keyboard Quick Reference List located in Appendix 2. The current chapter(Chapter 1) decribes the basic function of keys and their purpose. Using the Chapter 3 experiments, you can learn the keypad functions in depth by actual use. The Technical Appendices provide the ultimate level of detail. This is the information that reveals a computer's innermost secrets.

The keypad is organized into three logical groups of keys. They are the Interrupt Group Keys, the Data/Register/Flags Group Keys, and the Function Group Keys. What follows is a functional description of the keys in each of these groups.

1. Interrupt Group Keys -- RESET, BREAK, INTER

   a. RESET supplies a system reset to the Z80 microprocessor. This causes the Z80 to stop executing a program or HALT state and begin fetching and executing instructions that start at address 0000: the starting address of the Monitor program. The MT-80Z is initialized just as it is at power-up and will display READY on the 7-segment display. Pressing this key will not erase your program. It can, however, change your stack pointer and write some data at the high end of RAM (see Memory Map).

   b. BREAK: The BREAK key allows you to stop a program to examine registers and memory contents. Also, you can make changes to registers and memory and then restart the program from the point of interruption. It is an input to the Z80 NMI, Non-Maskable Interrupt. This is the equivalent to a CALL 0066. BREAK causes the MT-80Z to begin executing instructions at address 0066 in the Monitor program. If you press the key during execution of a program, the following happens:

      - The Z80 finishes the current instruction and stops executing your program.
      - The address display indicates the Program Counter contents at the time of program interruption.
      - The data display indicates the memory contents at the displayed address.
      - All Z-80 register contents are saved.
      - A check is made to see if your stack pointer is out of system stack area.

You cannot use the BREAK key to interrupt the Monitor. If you do, the warning SYS-SP occurs. The MT-80Z issues the SYS-SP warning when your program attempts to use the system stack area 1F9F to 1FAE. The user stack area begins at address 1F9E(See the MT-80Z Memory Map). If the MT-80Z is NOT running your program it is running the Monitor program and the stack pointer should be in the system stack area. The BREAK function "assumes" you are BREAKing a user program and always tests the stack pointer to be sure that it is 1F9E or lower. If BREAK is pushed when the Monitor is running, the stack pointer will be above 1F9E and cause the warning message.

c. INTER causes an execution of the lowest level of interrupt if certain conditions are met. INTER is connected directly to the INT pin of the Z-80. This is the Maskable Interrupt which, if enabled (unmasked), will cause a RST 38 to execute.

Here is what happens when you press INTER:

STEP 1: The Z80 begins fetching and executing instructions at Monitor Location 0038.

STEP 2: The Monitor routine at address 0038 uses the contents of RAM address 1FEE and 1FEF to form an address. This address is called a vector. When the MT-80Z is powered-up, the Monitor stores the data 66 at 1FEE and 00 at 1FEF. This would form the vector address 0066. This is also the interrupt address for the NMI BREAK key.

STEP 3: The MT-80Z begins executing instructions at the address formed by 1FEE and 1FEF.

The INTER interrupt may be enabled and its vector address set using the keyboard or instructions in your program. This enables you to set up maskable interrupt service software that is invoked by pushing INTER.

2. Data/Register/Flag Group Keys -- 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, AF, BC, DE, HL, AF', BC', DE', HL', IX, IY, SP, I IF, SZ H, PNC, SZ H', PNC', REG

The keys in this section are all dual purpose except for the REG key. When the numeric keys are used to type in hexadecimal numbers such as addresses and data, the key top indicates which key to use. When the REG key is pushed, the keys are reassigned to the job of viewing register contents and flags and the label above the key on the keyboard indicates which key to use.

3. Function Group Keys -- ADDR, DATA, PREV, NEXT, RELA, INSERT, DELETE, COPY, SET BRK PT, CLR BRK PT, LOAD, DUMP, PC, STEP, GO, USER.

a. ADDR, DATA, PREV and NEXT are the function keys used to enter and edit programs, examine memory contents, and modify registers.

(1) ADDR - This key signals that a memory address is to be input next. The 7-segment display shows memory address and data.

(2) DATA - This key signals that data for memory or registers is to be input next. It is normally pressed after the ADDR key. DATA is also used after the REG key to modify registers and flags.

(3) PREV - This key causes the displayed address to decrement to the previous address. The data display is updated each time the address is changed to reflect the contents of memory or registers at the new address. This key is used in the REG mode to back up to the previous register or flag displayed.

(4) NEXT - This key causes the displayed address to increment to the next address. The data display is updated each time the address is changed to reflect the contents of memory or registers at the new address. NEXT is used in th REG mode to advance to the next register or flag group displayed.

b. RELA provides valuable assistance when you write Z80 programs using JR (Jump Relative) and DJNZ (Decrement and Jump No Zero) instructions. These extremely powerful instructions consist of two bytes. The first byte is the operation code and the second is called a relative address. A relative address points to a location in memory by referring to its own location. The relative address must be a signed 2's complement number in the range +127 (7F Hex) to -128(80 Hex). When either the JR or the DJNZ instruction is executed, the address that it needs is formed by adding the relative address in the instruction to the instruction's own location in memory. Thus, the address formed is "relative" to wherever the instruction occurs in memory. When hand assembling Z80 programs, relative address calculation is a very difficult and error prone task.

The RELA key will calculate the relative address and store it in memory following the JR or DJNZ operation code. The only restrictions on the use of this key is that address calculation must be within MT-80Z RAM and not exceed +127 or -128.

34

c. INSERT and DELETE are program editing keys that make it easy for you to modify programs. One of the most frustrating errors in computing is to enter a long program and discover that an instruction was omitted by mistake. INSERT allows you to insert instructions or data into an already stored program. It does this by inserting at a specified address and moving up all the subsequent memory contents to the next highest address. DELETE works in the opposite fashion by removing one byte and "closing in" all the remaining memory contents to the next low address.

The range of memory operated upon by INSERT and DELETE is 1800 to 1DFF. In fact, anytime these keys are used the ENTIRE contents of this 1,535 byte address range is affected. For every DELETE key use, an 00 is stored in a high address starting with 1DFF. If you started at 1800 and pushed the DELETE enough times, the entire range of memory would be filled with zeros. INSERT pushes memory contents beyond 1DFF into outer space and they will be lost.

d. COPY allows you to "block move" large groups of memory contents from one address block to another. Such a move only copies memory and leaves the original source intact. This function comes in handy when you want to reuse part of your program's memory without losing a program. Using COPY, the program can be copied to a different area of memory, allowing you to enter new data and instructions in the area it previously occupied.

e. The SET BRK PT (Set Break Point) and CLR BRK PT (Clear Break Point) keys provide a powerful software debugging tool. Breakpointing allows you to execute a program one segment at a time. If a program fails to execute properly, it is often difficult to determine which segment is working properly and which segment is failing. By setting break points at strategic locations, you can execute a program segment by segment, observing register contents and flags in between segments and using this information to identify program faults.

f. LOAD and DUMP functions provide a means for using an audio cassette tape recorder to store and playback data or memory contents. Any block of memory can be named as a data file and the DUMP function transmits it serially to the record input of the tape recorder. Many files may be stored on a single cassette. When retrieval of the file becomes necessary, the LOAD function will ask for the file name and bring it in from cassette to memory.

Before using these functions, you must provide your own cassette recorder and cables. Suitable cables are available at Radio Shack and hi-fi stores. The EAR and MIC jack on the MT-80Z must be connected to the earphone output and microphone input of the recorder. The volume control of the recorder should be set to maximum.

g. PC allows you to set the user program counter register. The program counter is a 16-bit register that contains the address of the next instructions to be executed. When the MT-80Z is powered up, or RESET, one of the initializing procedures is to set the user's program counter to the lowest RAM address or 1800. It is assumed that you will enter your programs at starting address 1800. If you wish to start execution of your programs at another address, use the PC key to enter the new address.

The PC key is used mainly in conjunction with the GO key. For example, a program starting address is set using PC and then GO is pushed to start program execution.

h. GO is used either to start program execution at an address in the user program counter or to start the execution of certain keyboard functions.

In order to use the GO key, it is first necessary to have a valid address showing on the address/data display. The two main methods for doing this are: (1) ADDR key operation and (2) PC key operation. After the required address is properly displayed, a push of the GO key starts your program running.

If program execution is stopped with the BREAK key, you can resume by pushing GO. If you are using STEP to single step your program, GO can be used to start full speed execution.

i. STEP is similar to GO except that it executes only a single instruction for each key press. You can use ADDR or PC keys to establish the starting address for single stepping. The STEP function should not be confused with the hardware SINGLE CYCLE feature.

The STEP function is a powerful debugging tool which allows you to follow the "flow" of a program and discover where problems exist. The ability to view registers in between instructions will help you to understand how the Z80 microprocessor works.

If your program affects the Stack Pointer register so that it points to the system stack area (1FAF-1F9E), STEP will cause the warning SYS-SP on the address/data display. If the Stack Pointer is changed by your program to non-existent RAM, the warning Err-SP will be displayed.

j. The USER key is connected directly to pin 38 of the 8255 PPI chip. The 8255 is the large 40-pin chip located above the address/data display. On the 8255, pin 38 is bit 6 of port A. The MT-80Z has Port A mapped at I/O address 00.

In order to use the USER key, it is necessary to write a program that will input port 00 to the Z80 accumulator and test bit 6. If USER was not pressed during the input instruction, bit 6 will be a 1, otherwise it will be 0.

USER is a user-defineable key. With proper software, it can allow you to create a new function not presently available on the MT-80Z keypad.

PIO, CTC Expansion

The MT-80Z can be expanded to include two Z80 support chips: a PIO, Parallel I/O interface and a CTC, Counter Timer Circuit. They can be mounted in the space above the 7-segment display. The addition of these chips provides the following increased functions:

1. PIO - Two 8-bit I/O ports or one 8-bit bidirectional I/O port.

2. CTC - Four event counters or interval timers.

Connection to these functions is made using the 40-pin PIO CTC I/O BUS located next to the PIO socket area. The MIO-80, CTC/PIO Expansion Package for adding these chips is available from E & L Instruments, as E & L Part Number 200-8030.

Start

CHAPTER 2

GETTING STARTED WITH THE MT-80Z

Introduction

In this chapter, you are given detailed procedures for using the MT-80Z to perform some of the most used functions. These procedures will help you get acquainted with the general features of the MT-80Z and provide a starting place for learning to operate the computer. You will need an MT-80Z, the wall-mount adapter power unit supplied with the MT-80Z and, if possible, an audio cassette tape recorder. Even if a recorder is not available, you can still accomplish almost all of the objectives of this chapter of the User Manual.

Objectives

After successful completion of the activities in this chapter, you will be able to do the following:

1. Apply D.C. power to the FOX using the wall-mount adapter.
2. Enter and verify a Z80 program.
3. Use the REG function to modify a Z80 register pair.
4. Run a program.
5. Understand and use the following keys: ADDR, DATA, PREV, NEXT, RESET, PC and GO.
6. Use a cassette tape recorder and the DUMP and LOAD keys to save and load a program.

Power Up

Remove the top lid of the MT-80Z and set it up in front of you. There are many exposed electrical connections on the front panel of the computer. None of the voltages are high enough to cause electrical shock. However, if you use some conducting devices such as metallic pens, jewelry, tools or other lab equipment, take necessary precautions to avoid shorting the exposed electrical connections.

Locate the wall-mount adapter power unit supplied with the MT-80Z. Plug the connector into the POWER jack located in the top right corner. Now plug the adapter into a wall outlet. You should see the power up display scrolling to the left on the address/data display LEDs.

You will have the following display:

rEAd y

The MT-80Z is now "ready" for operation.

Your First Program

The program used in the following example will cause a 2kHz tone from the speaker. By proper use of the keypad functions, you will enter the program and then set the length of the tone.

Here is the program:

MEMORY ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	CD	CALL	CALL subroutine TONE2K
1801	E2	LO ADR	which expects HL to
1802	05	HI ADR	contain tone duration
1803	76	HALT	then halt.

The subroutine TONE2K is located in EPROM at address 05E2. It causes the speaker to beep at a 2kHz rate for a number of periods determined by the contents of register pair HL. The main use of this tone is for recording data on the audio cassette tape. We are going to "borrow" it for a short while for a different purpose.

Let's load the program and HL register pair. The keys used are ADDR, DATA, NEXT, PREV, REG and number/register keys.

STEP 1: Push ADDR. The speaker should beep (1kHz) indicating a key-press. The address/data display should light the decimal points (D.P.'s), in the address field indicating that an address may be keyed in. This is called the "address input mode".

STEP 2: Push number keys 1800. This is the address of the first byte of the program. The display will show 1800 in the address field. Our data field was F5, your data field could be different because the data is undefined at this time.

1.8.0.0. F5

STEP 3: Push DATA. The D.P.'s should shift to the data field indicating that data may be keyed in. This is called the "data input mode".

STEP 4: Enter the first byte of the program by pushing keys C and D. The display should look like this:

1800 C.d.

STEP 5: Push NEXT to enter the next byte of the program. The address will increment to 1801. Note that the D.P.'s still indicate that you are in the data input mode.

1801 F.5.

STEP 6: Enter the second byte, E2. The display now appears:

1801 E.2.

STEP 7: In similar fashion, i.e., pushing NEXT, then entering data, load the rest of the program. The displays for the remaining bytes are shown below:

1802 0.5.

1803 7.6.

STEP 8: In this step you will use the PREV key to view the previous memory contents. PREV is opposite in function to NEXT. Push the PREV key. You should see the previous address and memory contents: Continue with the PREV key until you are back to address 1800. By use of PREV and NEXT, you can verify memory contents to make sure that your program is correctly loaded. If you see the wrong memory contents on the data display, make sure you are in the DATA input mode and key in the correct byte.

STEP 9: The program is now loaded. Before running the program, it is necessary to set register pair HL. Start by pushing REG. The display shows:

rEG-

indicating that you may examine or change a register's contents, the "register view/alter mode". Now push the HL key. It has a 3 on top of the key and the letters HL above it. Don't use HL'. The data field now display HL and the address field, the current HL contents. No D.P.'s are lit.

STEP 10: Push DATA. The D.P.'s indicate the register that is in the alter mode. In this case, it is the L register. Our display looked like this, but yours may be different:

FdF.d. HL

Now press the 0 key. Note that both digits with D.P.'s changed to zero even though you pressed only a single key. The entire 8 bits of the L register are now zero.

STEP 11: Push NEXT. Did you see the D.P.'s shift to the left? Press the 4 key. H now contains a 04. The HL register pair now contains 0400 as shown by the display:

0400 HL

STEP 12: The PREV key also works in the register view/alter mode. Go ahead and push PREV. The results should be opposite that of the NEXT key. The D.P.'s should be shifted to the right. You could change L to something else now if you wanted to.

Now that the program and registers are loaded, it's time to run the program and see (hear) the results. For this procedure, the PC, GO and RESET keys are used.

STEP 13: Push PC. The display should indicate:

1800 Cd

or a program counter value of 1800. This is the starting address of the program. Try pressing the ADDR and DATA keys alternately. You should see the D.P.'s moving back and forth from the address field to the data field.

STEP 14: Push GO. You should hear the key press tone, then a longer 2kHz tone. The display will be blank and the HALT LED lit. The Z80 HALT condition was caused by the last instruction of the program. A "HALT"ed Z80 cannot provide segment and digit outputs for the display or read the function keys. Try pressing some of the function keys. The MT-80Z appears to be inoperative. One key will bring it back to life: RESET.

STEP 15: Push RESET. What happens? You should see the familiar power on "rEAdy" display and the HALT LED off. The RESET key is directly wired to the Z80 and cannot be ignored by the HALT condition. The MT-80Z is now under control of the Monitor program and is waiting for you to press a key. Press PC, GO and RESET a few more times to get some practice running a program.

Some Experimentation

Let's try a few different values of HL and observe changes in the tone duration. With your program still in memory, use the REG key to change HL to 1000. Use the same procedure shown in Steps 8, 9 and 10 for making the change. Run the program. How long is the tone? We clocked it about two seconds.

Now try HL=2000. The tone should be about four seconds. Here is a hexadecimal arithmetic problem for you. The displayed HL data is a hexadecimal number. If 1000 causes a 2 second tone, what HL contents will cause a 1 second tone? Try your calculation by entering the HL value and running the program. A chart follows which gives some HL values with the corresponding time for the tone duration.

HL VALUE	TIME	(approx.)
0400	.5	seconds
0800	1	second
1000	2	seconds
2000	4	seconds
0000	15	seconds

Note that the smallest number (0000) gives the longest time. The subroutine TONE2K actually "counts down" or decrements the HL register pair to determine tone duration. When the value 0000 is decremented once, the following result is obtained:

$$\begin{array}{r} 0000 \\ -\quad 1 \\ \hline FFFF \end{array}$$

The resulting high number, FFFF, causes the long duration.

Before you leave this section, try the values in the table above and then some of your own.

Using the Tape Recorder

What you will need for this section is the MT-80Z, a standard audio cassette tape recorder, blank cassette and at least one cable fitted with a miniature phone plug at one end and, at the other end, a connector that will mate to the cassette recorder MIC input and SPEAKER output. It is very likely that a cable with a miniature phone plug on each end will suffice.

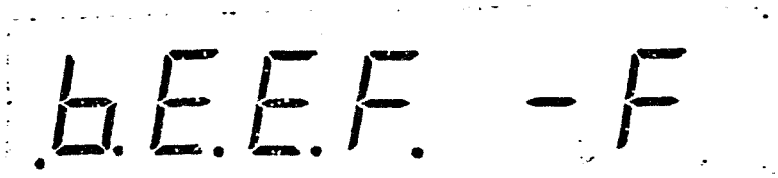
The steps that follow outline the procedure for storing (DUMP) and loading (LOAD) a program on audio cassette tape. It is assumed that you still have in memory the program listed at the beginning of this chapter. If not, refer to Steps 1 through 7 and load the program.

STEP 1: Connect an audio cable from the tape recorder MIC input to the MT-80Z MIC jack located at the top right corner.

STEP 2: Install a cassette tape in the recorder and advance the tape BEYOND THE LEADER. Leaderless tape cassettes are available and are especially designed for audio cassette tape computer data storage. Use a good quality of tape if you want to reduce the likelihood of errors in your stored programs. Most general purpose tapes have varying amounts of leader. If the recorder has a counter, push the counter reset button so that all zeros are showing.

STEP 3: Push DUMP. The data display indicates "-F" which is how the FOX asks you the file name for the data to be recorded on tape.

STEP 4: The file name can be any 4-digit hexadecimal number from 0000 to FFFF. Let's use BEEF; a hex number that is also a word. Using the number keys, enter the file name BEEF. The display should look like this:



STEP 5: Push NEXT. The "-S" in the data display is for starting address. Our program starts at address 1800, so enter: 1800

1800 -S

STEP 6: Push NEXT. The last question the FOX asks is the ending address ("-E") of the block of memory to be stored on tape. Our program ends at address 1803. Enter 1803 to get the following display:

1803 -E

STEP 7: Your input to the tape DUMP function has established the following:

a) File name -- BEEF
b) Starting address -- 1800
c) Ending address --1803

STEP 8: Start the tape recorder in the RECORD mode and after the tape is rolling, push GO. If your tape recorder has an adjustable record level, you may have to experiment a few times to make a proper recording. Most cassette recorders have an automatic level control.

STEP 9: You will hear some tones from the speaker. After the tones stop and the display shows:

1803 76

Stop the recorder. If all has gone well, the file, BEEF, has been saved on tape.

STEP 10: Let's test the recording. Rewind the tape to the starting place or 000 if you have a counter. Now change the cable from MIC to EAR on both recorder and computer.

STEP 11: Push LOAD. The MT-80Z now asks for the file name:

b.E.E.F. -F

indicates that the MT-80Z retained the file name from the last DUMP operation. The D.P.'s indicate a file name input mode. BEEF is the correct file name, so no change is necessary.

STEP 12: Push GO and start the tape recorder in the play mode. The display will change indicating each stage of the tape LOAD process.

- First process: wait for the "mark" tone

. . . . . .

- Second process: file name found

bEEF -F

- Third process: loading file

- - - - - -

o Fourth process: file loaded correctly

1803 76

If your results for the last few steps are not the same as those given, you may be experiencing some tape problems. Here are a few facts that may help you track down some possible problems.

a. Incorrect record level control. Automatic record level control, sometimes called ALC is ideal for computer use. The small, inexpensive cassette recorder usually has this feature.

b. Playback volume must be nearly maximum.

c. If you hear the tones played back through the MT-80Z speaker, they are being properly received at the EAR input. The audio going into the MT-80Z is NOT DIRECTLY COUPLED TO THE SPEAKER.

d. Watch out for tape leader!

If your results match those of Step 12 above, you probably have a good recording. The real test is to temporarily remove power from the MT-80Z to "lose" the program from the volatile RAM. Then with power connected, repeat steps 10 through 12 to LOAD the program from tape. This time, you will have to enter the file name "BEEF" in step 11. Don't forget to set HL before running the program.

CHAPTER 3
EXPERIMENTS

Introduction

This chapter contains 13 experiments designed to give you hands-on experience with the MT-80Z. The previous chapter started you off with some general functions. The experiments in this chapter cover the remaining keyboard functions, I/O ports and SINGLE CYCLE operation.

Most of the experiments require only the MT-80Z and the wall-mount adapter power supply. Some of the experiments require a length of #22 or #24 solid hook-up wire. Prior to starting each experiment, be sure to read the section in Chapter 1 that describes the keys or functions used.

Experiment List

1. REG key - viewing and changing Z80 registers and flags
2. INSERT and DELETE keys - program editing
3. COPY key - block transfer of memory contents
4. RELA key - relative address calculation for DJNZ and JR
5. STEP key - single instruction stepping
6. BRK PT keys - setting and clearing break points for program debugging
7. BREAK key - stopping and restarting a program without losing registers and stack
8. INTER key - maskable interrupts
9. USER key - defining your own keyboard function
10. Speaker and TONE LED - sound from the MT-80Z
11. Logic Indicators - Port 1 and 2 LED displays, PORT and BUS SOCKETS
12. Logic Switches - Port 1 and 2 logic switches and the PORT and BUS SOCKETS
13. SINGLE CYCLE operation - single stepping by machine cycle and data bus monitoring

Objectives

After successful completion of the experiments in this chapter you will be able to do the following:

1. Understand and use the following keys: REG, INSERT, DELETE, COPY, RELA, STEP, SET BRK PT, CLR BRK PT, BREAK, INTER, USER, CYCLE-RUN (S3) and PB1, the single cycle stepping switch.

2. Use the PORT and BUS SOCKETS to connect an I/O device to the MT-80Z.

3. Use the PORT LEDs and logic switches (S1 and S2) to perform I/O operations.

4. Use the SINGLE CYCLE switches to step a program and monitor the results of each cycle on the PORT 1 LEDs.

EXPERIMENT 1 - REG KEY

The purpose of this experiment is to help you learn how to use the REG function to observe or modify any of the MT-80Z registers or flags.

This function can be used when you want to do any of the following:

1. Set the value of any register prior to running a program.

2. Set or clear any of the flags prior to running a program.

3. Observe the results in the registers after a program has been run.

4. Observe the flags after running arithmetic or logical instructions.

5. Monitor flag and register results between instructions while single stepping using the STEP key.

<u>STEP 1</u>: Apply power and press the REG key. The rEg- indicates that all the number keys are now register and flag keys.

<u>STEP 2</u>: Select AF by pushing the AF key. Now, you will see the letters AF in the data display and the register pair (accumulator and flags) contents are shown in the address field. These two steps are used when you wish to view register pair contents.

<u>STEP 3</u>: Let's look at the rest of the registers. Push BC, DE, HL. What happens in the data display? Now push AF'. This is the first of the alternate set of Z80 registers. Look closely at the data display. You will see a decimal point (D.P.) indicating the "prime" or alternate AF.

STEP 4: Look at the rest of the alternate register set: BC', DE' and HL'. There will be a D.P. indication on all these.

FdFd HL.

STEP 5: Now select IX. It is difficult to make an X using the seven segment display. Here is how the MT-80Z does it:

FdFd 1h

STEP 6: Push IY. It's easier to display a Y isn't it? Now push SP. The stack pointer (SP) has been set by the monitor at power-up to the address displayed.

1F9F 5P

STEP 7: Push I-IF. The display format is shown below:

0000 1F

I V IFF

The left two digits display the Z80 I register. This register is used for certain MODE 2 interrupts to form the high 8 bits of the interrupt vector address. The right-hand digit (least significant digit) of the address display shows the status of interrupt flip-flop IFF2. This is an interrupt mask. When it is 0, maskable interrupts are enabled. The display shown above indicates interrupt vector high address 00 and interrupts disabled. Anytime you apply power or push RESET, the Monitor program establishes this condition.

STEP 8: Push SZ•H. This is the first of the flag keys. The display shows the four high order bits of the flag register. From left to right: They are: Sign, Zero, Unused bit, Half carry.

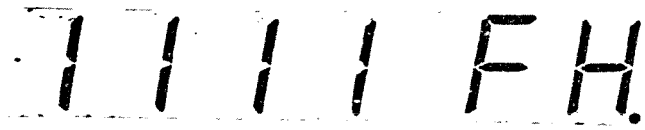
1 1 1 1 F H

The data display shows FH for "Flag High". The "ones" indicate that the S, Z and H flags are set. The flag display is binary. Now press •PNC. What do you see?

The FL or "flag low" display format is (left to right) unused bit, parity, N (subtract flag) and carry.

STEP 9: Push SZ•H'. What is different about the data display?

The D.P. in the data display indicates the alternate flag set.



Now try •PNC'. Again, the D.P. indicates the alternate flag set.

STEP 10: The DATA key is used to modify a register. Let's modify the F register of register pair AF and check the flags to see if they change. Push AF then DATA. What do you see?

The D.P.s indicate the F register is in the modify mode. The data/register keys are now used to enter data. Push the 0 key. What did you observe?

The entire F register is now "zeroed" with only one press of the 0 key. The leading zero is automatically entered. Now, check flags SZ•H and •PNC to verify the zero state of the flags. Do this by pushing REG then SZ•H and •PNC keys. Does the FH and FL display agree with the change made in the F register?

Both FH and FL should indicate all zeros.

STEP 11: How can you modify the entire register and flag set?

By use of NEXT (and PREV), you can easily step through the registers and modify contents. Start with AF and change every register to 00. Push REG, AF and DATA. Push 0 to change the F register. Push NEXT. Did you see the D.P.s shift?

Push 0 and NEXT again. What did you observe?

The next register pair in line is BC. The D.P.s indicate the C register is ready for modification. Continue with 0 and NEXT until you have completed I•IF. It is not necessary to zero the flags. Do you know why?

The flags were zeroed when AF and AF' were modified. Check them by using NEXT.

STEP 12: Continue to check each register pair by pushing NEXT. Are they all zero?

All registers should display zeros.

STEP 13: Push RESET. Check the SP register. Is it still zero?

Whenever RESET is pushed, the monitor program automatically resets the SP to 1F9F. Check the other registers. What do you find?

All the other registers should be zero.

EXPERIMENT 2 - INSERT AND DELETE KEYS

The purpose of this experiment is to help you learn how to use INSERT and DELETE to change memory contents. You can use these keys to insert added instructions or delete unwanted instructions from your programs. These keys will save considerable time when you need to modify a long program.

STEP 1: Enter the following test data into memory. This list of numbers is not a program. The numbers will make it easy to recognize the inserting and deleting operations.

ADDRESS	DATA
1800	11
1801	22
1802	33
1803	44
1804	55
1805	66
1806	77
1807	88
1808	99
1809	AA

In the steps that follow, you will insert the number FF at address 1804 and "push" the remaining numbers to the next highest address. The result of the operation should look like this:

ADDRESS	DATA	
1800	11	
1801	22	
1802	33	
1803	44	
1804	FF	← inserted
1805	55	
1806	66	
1807	77	
1808	88	
1809	99	
180A	AA	

STEP 2: Use ADDR and select address 1803. This address precedes the address to be inserted. The INSERT function moves up to the insert address.

STEP 3: With address 1803 displayed, push INSERT. What do you observe?

Address 1804 is displayed with contents set to 00. What do the D.P.s indicate?

The MT-80Z is ready for the new data. It has already moved the test data from 1804-1809 to 1805-180A. A 00 has been inserted at address 1804.

STEP 4: Enter the FF. The inserted 00 has been changed to FF. This completes the insert operation. What would happen if the FF was not entered?

The test data would show a 00 had been inserted.

STEP 5: Use ADDR and NEXT to verify the insert operation. What do you find?

Your check of address 1800-180A should match the table above.

STEP 6: How would you insert a DB at address 1800?

The procedure is to select the previous address 17FF, use INSERT to move up to address 1800 and enter BB. Try this procedure. Do your results match the listing below?

ADDRESS	DATA	
1800	BB	← inserted
1801	11	
1802	22	
1803	33	
1804	44	
1805	FF	
1806	55	
1807	66	
1808	77	
1809	88	
180A	99	
180B	AA	

In the next 2 steps, you will delete the inserted BB and FF to restore the test data to the condition at the beginning of the experiment.

STEP 7: To delete the BB at address 1800, first use ADDR to select address 1800. Next, push DELETE. What do you see?

Address 1800 contains 11. Use NEXT to check the remaining test data. What are the results?

ADDRESS	DATA	
1800	11	BB was deleted
1801	22	
1802	33	
1803	44	
1804	FF	
1805	55	
1806	66	
1807	77	
1808	88	
1809	99	
180A	AA	
180B	--	

STEP 8: How would you delete the FF now at address 1804?

Select address 1804. Push DELETE. Check memory contents 1800-1809. What did you find?

The table should be in its original condition.

10

STEP 9: What happens if you try to delete the memory contents of address 05FE?

Nothing happens. Address 05FE is ROM and cannot be changed. The display will not indicate any changed data in memory.

STEP 10: What happens if you try to insert data at address 17FE?

Again, nothing is changed. The display address, 17FE did not increment to the next address as before. This is an indication that you are trying to insert data in ROM and beyond the range of the INSERT and DELETE functions.

The range of INSERT and DELETE is 1800-1DFF. Each time you use these keys, this entire range of memory is affected.

NOTE: If you have more than one program in memory, an INSERT or DELETE affects the other programs.

In the next steps, you will prove the memory range of INSERT and DELETE.

STEP 11: Store the following test data in memory:

ADDRESS	DATA
1DFB	11
1DFC	22
1DFD	33
1DFE	44
1DFF	55
1E00	66
1E01	77

STEP 12: Insert an FF at address 1DFD and check the memory contents. What do you find?

The previous contents, 55, of 1DFF were not moved up to 1E00.

ADDRESS	DATA	
1DFB	11	
1DFC	22	
1DFD	FF	
1DFE	33	
1DFF	44	
	lost	
1E00	66	
1E01	77	unaffected

STEP 13: Now, delete the FF at address 1DFD. How has address 1DFB-1E01 been affected?

ADDRESS	DATA	
1DFB	11	
1DFC	22	
1DFD	33	
1DFE	44	
1DFF	00	inserted
1E00	66	
1E01	77	unaffected

The delete function inserts 00 at 1DFF each time DELETE is pressed. Address 1E00 and beyond are unaffected.

12

EXPERIMENT 3 - COPY KEY

The purpose of this experiment is to help you learn how to use the COPY function. COPY will "block move" a copy of memory contents from one section of memory to another.

STEP 1: Load the test data shown below. This is not a program, but easily recognized data to experiment with the COPY function.

ADDRESS	DATA
1800	00
1801	11
1802	22
1803	33
1804	44
1805	55
1806	66
1807	77
1808	88
1809	99

In the next 8 steps, you will block move a copy of memory contents from addresses 1800-1809 to address 1900-1909 and verify the operation.

STEP 2: Push COPY. What do you see?

The -S in the data display is a prompt. This means the MT-80Z is "asking" for the starting address of the block of memory to be copied. The D.P.s indicate the address input mode.

STEP 3: Enter 1800 as the starting address. The display should look like:

STEP 4: Push NEXT. What happens?

The data display shows the next prompt: -E, the ending address of the block of memory to be copied.

STEP 5: Enter 1809. So far, the COPY function "knows" the starting and ending address of the test data.

STEP 6: Push NEXT. What is this prompt "asking"?

The prompt, -d, is for destination starting address.

STEP 7: Enter the destination address 1900.

STEP 8: Push GO to complete the copy operation. What do you observe on the display?

What is displayed after the GO key is pressed depends on whether the copy was toward a higher address or a lower address. In our case, copy was from low to high and the display shows the starting address of the destination and the contents:

1900 00

STEP 9: Use NEXT to verify the copy of the test data. What did you find?

An exact copy of the test data should be stored at address 1900-1909.

In the remaining steps, you will block move a copy of memory from address 1900-1909 to address 1800-1809 and verify the operation.

STEP 10: Change the contents of 1900-1909 to the following:

ADDRESS	DATA
1900	AA
1901	BB
1902	CC
1903	DD
1904	EE
1905	FF
1906	10
1907	20
1908	30
1909	40

STEP 11: Push COPY and answer the -S prompt by entering the starting address 1900.

STEP 12: Push NEXT and answer the -E prompt by entering 1909.

STEP 13: Push NEXT and answer the -d prompt by entering the destination address 1800. The COPY function "knows" that we want to copy memory address 1900-1909 to 1800-1809.

STEP 14: Push GO. What do you see. Is it the same display the last GO operation?

The display should look like:

1809 40.

This is the last address of the copied block of memory. Also, the data display shows the contents at that address. This is a normal display when copying from higher addresses to lower addresses.

EXPERIMENT 4 - RELA key

The purpose of this experiment is to help you learn how to use the RELA key to calculate the second byte of the Z80 JR (Jump Relative) and DJNZ (Decrement and Jump No Zero) instructions. The second byte of these instructions is often referred to as the relative address, relative offset, or displacement. The format for these instructions is shown below:

INSTRUCTION CODE	MNEMONIC		COMMENTS
18	JR dis	Jump relative	
38	JR C, dis	Jump relative if carry true	
30	JR NC, dis	Jump relative if carry not true	
--			
28	JR Z, dis	Jump relative if Z flag true	
--			
20	JR NZ, dis	Jump relative if Z flag not true	
--			
10	DJNZ dis	Decrement B and jump if B $\neq$ 0	
--			

In the mnemonic, dis refers to displacement. The displacement is a 2's complement signed number ranging from a decimal +127 to -128. In hexadecimal, the range is from 7F to 80. The MT-80Z uses the second byte of the jump instruction to determine the destination address of the jump. After the displacement (second byte) of the instruction has been fetched, the Z80 replaces its program counter (PC) contents with program counter + displacement. The new PC value causes the jump. These instructions are used often because they are relocatable, i.e., the instruction does not contain an address. In contrast, JP label is a 3-byte instruction which contains an absolute address as the last two bytes. This instruction is written to operate at only one memory location.

Consider the following program:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	3E	LD A, 00	A = 0
1801	00		
1802	3C	INC A	A = A+1
1803	D3	OUT (FF),A	Output register A to port FF
1804	FF		
1805	18	JR dis	Jump relative to address 1802
1806	--		

This program sets A to zero, then continues to increment and output. The effect, if seen, is a high-speed binary counter on port FF. the second byte of the JR dis is missing and needs to be calculated before the program can be completely loaded.

In the steps that follow, you will calculate the second byte (dis) of the JR instruction using RELA. The program will not be run. Don't despair! There are "flashy" programs in some of the other experiments.

STEP 1: Load the program up to, and including address 1805.

STEP 2: Push RELA. Do you see the -S? This is a prompt "asking" you to enter the source address or the address of the JR instruction.

STEP 3: Enter the 1805. Now push NEXT. What happened in the data display?

The -d is a prompt "asking" for the destination address of the jump. What is this address? In order to keep incrementing we must jump each time to address 1802.

1.8

STEP 4: Enter 1802. Now, it's time to calculate the displacement. Do this by pushing GO. What is your observation?

The display should look like this:

1806 F.b.

Here is what happened. RELA calculated dis as FB and stored it in memory directly after the JR instruction. Real service! Your program is now complete.

STEP 5: The range of the jump is limited. Let's try to calculate a displacement that is out of range to see what happens. We will assume the JR instruction is located at address 1900 and the destination address of the jump is 1800. This is a 256 byte distance. Is this out of range?

STEP 6: Push RELA and enter the source address 1900.

STEP 7: Push NEXT and enter the destination address 1800.

STEP 8: Push GO to calculate. What happened?

-Err

We tried to go too far. -Err is always displayed when the range for these instructions is exceeded.

EXPERIMENT 5 - STEP KEY

The purpose of this experiment is to learn how to use the STEP key for single instruction stepping of a program. Computer programs usually run automatically at CPU clock speed. Single stepping slows the computer to allow you to view I/O operations and changes in register contents as they happen. Also, you can trace the sequence of instructions being executed in complicated programs full of conditional instructions. Single stepping makes it easier for you to locate program faults (bugs).

In this experiment, you will load and execute a simple program using both single instruction stepping (STEP) and full C P U - speed (GO). Also, you will learn how to set up a simple output port using the PORT 1 LEDs. You will need a short length of #22 or #24 solid hook-up wire.

STEP 1: Apply a jumper between PORT SOCKET connections OUT FF and P1CL. The jumper connects the Port 1 clock input (Port 1 LED interface) to the output address control for port FF. The jumper "maps" the Port 1 Logic Indicators to port address FF. The Port 1 Logic Indicators are permanently interfaced to the data bus. No additional wiring is needed.

STEP 2: Load the program listed below. It should look familiar if you have done Experiment 4.

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	3E	LD A,00	A = 0
1801	00		
1802	3C	IN A	A = A+1
1803	D3	OUT (FF),A	Output register A to Port FF
1804	FF		
1805	18	JR dis	Jump relative to address 1802
1806	FB		

STEP 3: What is the function of this program?

The first instruction sets A to 00. The remaining instructions increment A, output A to port FF, then loop back to continue incrementing and outputting. The jumper on the PORT SOCKET causes the Port 1 Logic Indicators to latch and display register A after every OUT (FF),A instruction.

STEP 4: In this step, you will run the program at full speed. Push PC, then GO. What do you see on the display? What do you see on port FF (Port 1 Logic Indicators)? Can you explain why?

The program instructions do not include OUT instructions for the 7-segment displays. Therefore, it is normal for them to be blank while your program is running. All the Port 1 LEDs appear to be on because the program is running at high speed. The incrementing and outputting is too fast to observe.

STEP 5: Stop execution of the program by pushing the RESET key. Can you explain the appearance of the Port 1 LEDs?

The LEDs display the register A contents last output before the RESET key was pushed. In the steps that follow, you will single step the program.

STEP 6: Push PC and observe the 7-segment display. What is the result?

72

The address 1800 is the first address of the program and the 3E in the data display is the code for the first instruction.

STEP 7: Push STEP. What do you see on the 7-segment display?

When STEP was pushed, the first 2-byte instruction LD A,00 (3E00) was executed. The display shows the next address and instruction to be executed: 1802 3C.

STEP 8: Push STEP. What is displayed?

1803 D3

STEP 9: Push STEP. What is displayed?

1805 18, the JR dis instruction. Can you predict the next instruction and address displayed after one more STEP?

STEP 10: Push STEP. What is displayed?

1802 3C is the destination of the relative jump instruction.

STEP 11: Why didn't the STEP key step one address each time it was pushed?

Z80 instructions can be one to four bytes in length. STEP is single instruction stepping. The change in address will depend entirely upon the length of the instruction stepped.

STEP 12: So far, we have been concentrating on the 7-segment display. Let's turn our attention to the Port 1 Logic Indicators wired as output port FF. Note the number displayed (in binary) on the output port LEDs: _____

STEP 13: Push STEP to execute the INC A instruction. The next instruction to be stepped is OUT (FF),A. What should happen to the port LEDs after the next STEP?

STEP 14: Push STEP. What happened?

The binary display incremented by one. This is proof the OUT (FF),A instruction was executed.

How many times do you have to push STEP to increment the port display?

STEP 15: Continue to push STEP and determine the number of instructions or STEPs for each change in the port display.

The program loop consists of three instructions: INC A, OUT (FF),A, and JR dis. The answer is three pushes.

STEP 16: When using STEP, you can view and change registers and flags between steps. The port LEDs and the A register should contain the same number. Use REG and AF keys to display the AF register pair. What do you see?

STEP 17: The A register (accumulator) value should match the port LED display. Push DATA then NEXT to place the A register in the input mode. Set A to 00 by pushing the 0 key.

STEP 18: Resume stepping by pushing PC, then STEP. Continue to push STEP until the port LEDs change. What is the new port display?

The 00000001 displayed is due to the "zeroed" accumulator being incremented and output to port FF.

EXPERIMENT 6 - BRK PT KEYS

The purpose of this experiment is to help you learn how to use the SET BRK PT and CLR BRK PT keys to set and clear program breakpoints. A breakpoint is an address in a program where you want to temporarily stop execution. When execution is stopped, you can view and change register contents and flags. After observations and changes are made, program execution can resume from the breakpoint by using STEP or GO.

A breakpoint is useful for debugging (correcting) a program. If a long program fails to execute properly, a good strategy is to "divide and conquer". A breakpoint can be placed in the middle of a program to stop execution. When stopped, registers, flags and I/O ports can be examined to verify program operation. If the first program segment checks okay, then place a breakpoint in the middle of the untested segment. By continuing this process, you can reduce the size of program segment being debugged.

In this experiment, you will load a simple program and place a breakpoint to help observe and verify program operation. Each time the breakpoint stops the program, you will observe registers and an output port. You will need a short length of #22 or #24 solid hook-up wire.

<u>STEP 1</u>: Connect a jumper between PORT SOCKET connections OUT FF and P1 CL. This step is the same as STEP 1 of Experiment 5. The jumper "maps" the Port 1 Logic Indicators to port address FF.

STEP 2: Load the following program:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	3E	LD A,00	A = 0
1801	00		
1802	47	LD B,A	B = A
1803	2F	CPL	A = $\overline{A}$
1804	D3	OUT (FF),A	Output A to port
1805	FF		FF
1806	3D	DEC A	A = A-1
1807	04	INC B	B = B+1
1808	18	JR dis	Jump relative to
1809	FA		address 1804

STEP 3: What is the function of this program?

The first two instructions set registers A and B to 00. The CPL instruction at address 1803 compliments the A register. The next four instructions form a loop which has the following continuous operation: output A to the Port 1 Logic Indicators (port FF), decrement A, increment B, then jump back to output.

STEP 4: Push PC, then GO. What do you see on the 7-segment display? What is the appearance of the Port 1 Logic Indicators (port FF)?

This program does not output to the 7-segment display. It is normal for the 7-segment display to be blank. The port LEDs appear to be steadily lit. Are they?

78

No. The A register (accumulator) is continually being decremented to the output. The speed of program execution is too fast for you to observe what is actually happening.

In the next four steps, you will set a breakpoint in the middle of the program address 1804, and observe the effects of the first four instructions.

STEP 5: Select the breakpoint address: push ADDR, then enter 1804.

STEP 6: With address 1804 displayed, push SET BRK PT. What changed on the display?

The MT-80Z indicates a breakpoint address and instruction code by lighting all six D.P.s.

1.8.0.4. d.3.

STEP 7: Now that the breakpoint is set, let's run the first four instructions. Push PC to display the starting address: 1800. With the address, 1800 displayed, push GO. What do you observe on the 7-segment display? Which instructions have been executed?

When the program stops (breaks) the display shows the next instruction TO BE EXECUTED. The display should indicate 1806 3.d. This is the address and first byte of the FIFTH instruction. The first four have been executed.

STEP 8: Use the REG key to check registers A and B. What do you find?

A = FF and B = 00. What do the Port 1 LEDs (port FF) display?

We observed a binary 11111111. This agrees with the current register A contents. By use of the REG function and observation of the port LEDs, you have verified the program operation up to AND INCLUDING the breakpoint.

In the next three steps, you will observe the operation of the instructions in the loop: OUT (FF), A, DEC A, INC B, and JR dis and back to OUT (FF), A. Note that the breakpoint is in the loop. This means that each press of the GO key will cause the MT-80Z to make one pass through the loop.

STEP 9: Press PC. The display should be 1806 3.d. If it is not, use the ADDR key and enter 1806. This will be the first instruction after the break.

STEP 10: Push GO. What do you observe on the 7-segment display and the Port 1 Logic Indicator?

The 7-segment display did not change. Do you know why? Hint: observe the Port 1 LEDs.

When you pressed GO, the following instructions were executed: DEC A, INC B, JR dis and OUT (FF),A. The program stopped and the MT-80Z is waiting to execute the DEC A again. The Port 1 LEDs (port FF) now displays 11111110; a value one less than you had before GO was pushed.

STEP 11: Continue to push GO and observe the changes on the Port 1 Logic Indicators. What do you see?

We observed the port LED display decrementing. Breakpoints must be set at the address of the FIRST BYTE of the instruction code.

In the next step, you will observe the result of setting a breakpoint at an incorrect address.

STEP 12: Change the breakpoint to address 1805. This address holds the second byte of the OUT instruction. If the display is 1806 3.d, push PREV to view 1805 F.F. and push SET BRK PT. If the display is other than 1806 3.d., use ADDR and enter address 1805. The old breakpoint is removed. Push NEXT to prove it. What do you see?

There are D.P.s on the data field only. The breakpoint has been removed.

STEP 13: Push GO. What do you observe on the 7-segment display and port LEDs? Why?

The incorrectly set breakpoint is not functioning. The program is executing the loop at high speed.

STEP 14: Breakpoint removal is done using CLR BRK PT (clear breakpoint). Use the ADDR and number keys to select the breakpoint address: 1804. When selected, you should see all D.P.s lit, indicating the breakpoint.

STEP 15: Push CLR BRK PT. What do you see?

The display, F.F.F.F. F.F. indicates breakpoint removal. Use ADDR to select 1804 again. Do you see proof of breakpoint removal?

Only four D.P.s are lit, indicating a normal address input mode.

Here are rules governing the use of breakpoints with the MT-80Z:

1. Only one breakpoint may be set at a time.

2. When the program is stopped, you may observe and change registers.

3. Breakpoints cannot be set in ROM.

4. Breakpoints must be set at the address of the FIRST BYTE of the instruction code.

5. Breakpoint addresses are displayed by lighting all six D.P.s.

6. Breakpoint removal is verified by the display F.F.F.F. F.F.

02

EXPERIMENT 7 - BREAK KEY

The purpose of this experiment is to learn how to use the BREAK key to stop program execution at any time to observe and change registers and flags. Also, you will learn how the BREAK key differs from RESET key operation.

In this experiment, you will load a program, use BREAK to stop execution, change register contents and resume execution. Also, you will observe the difference in using the BREAK and RESET keys.

STEP 1: Load the following program:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	CD	CALL TONE2K	Call a subroutine in
1801	E2		the monitor at address
1802	05		05E2
1803	76	HALT	after the subroutine is completed, halt.

This is the same program used in the Chapter 2 familiarization experiment. Do you recall the function of this program?

This two-instruction program calls a subroutine, TONE2K, stored in the MONITOR ROM, executes it and halts. TONE2K uses the value stored in HL to determine the duration of the tone from the speaker. In Chapter 2, you changed HL values and observed tone duration. Here is some new information regarding TONE2K: register C contains the value used to determine the tone frequency.

STEP 2: Use the REG function and set HL = 0000. Remember from Chapter 2: HL = 0000 gives the longest tone duration.

STEP 3: Push PC. What is displayed?

1800 c.d., the user P.C. is the starting address of the program.

STEP 4: Press GO, then BREAK. You should have let the tone beep for a few seconds. What do you observe on the display. Why is the address displayed not in User RAM?

After pressing BREAK, the program in execution is interrupted and the user Program Counter (P.C.) and memory data is displayed. This displayed address is "pointing" to the next instruction to be executed when you resume (GO) from the BREAK. The address displayed will be in the range: 05EA to 05F3. This range is the ROM subroutine loop that generates the tone. There is no way of knowing exactly where you "froze the clock" by pushing BREAK. We observed 05Ed 1.0.

STEP 5: Use the REG function to view HL, C and SP. Note the values: HL = _____, C = _____, SP = _____. Can you explain these values?

a 4

We observed the HL value, 6D2E. The subroutine, TONE2K, was counting this value down toward zero until you pressed BREAK. If the tone stops and the MT-80Z halts (HALT LED), HL has reached zero. You should observe the value, 1F in C. This value was loaded by the subroutine and corresponds to the frequency, 2kHz. The Stack Pointer contains the value 1F9D. The CALL instruction of your program caused the Z80 to "stack" the return address. To view the stack contents, push ADDR. The contents of stack address 1F9D is 03. Push NEXT. The contents of stack address 1F9E is 18. The return address is 1803.

STEP 6: Change the contents of register C to FF.

STEP 7: Push PC, then GO. Listen to the tone for a few seconds then push BREAK. What did you hear? Why?

The lower tone is due to the larger value in C. The tone frequency is related to C by the following:

Frequency = 1/2 ((44+13C) x clock period)

For C = 1F and a clock period of 559 nS:

$F = 1/2\ ((44+(13)(31))\ 559 \times 10^{-9})$

F = 2001 Hz

What is the tone frequency for C = FF?

266Hz.

STEP 8: Change C for a frequency higher than 2KHz. Which value did you choose? C = _____ Frequency = _____.

STEP 9: Push PC, GO, then BREAK to verify the higher frequency. We chose 0F for a frequency above 5KHz. If your tone has stopped already, push RESET, PC, GO, then BREAK.

STEP 10: What is the value of HL? It should be lower than the value noted in Step 5.

We indicated a value of 20DE at this step.

STEP 11: Push PC, then GO and let the tone run until it stops. It stops when HL has been decremented to zero. What is the state of the HALT LED?

The HALT LED lights after the Z80 executes a HALT instruction. There are two methods of recovering from a HALT condition:

1. Press BREAK
2. Press RESET

STEP 12: Press BREAK. What happened to the HALT LED? can you explain what is shown on the 7-segment display?

When you press BREAK, the HALT condition is removed. The 7-segment display shows 1804 F.5. After the HALT instruction at address 1803 has been executed, the user P.C. "points" to the next memory address.

ab

STEP 13: Use the REG function to check the contents of HL. What do you observe?

HL = 0000.

STEP 14: What is the value of the Stack Pointer (SP)?

After the subroutine has been executed, the return address, 1803, (see Step 5) has been "popped" off the stack and the Stack Pointer returned to 1F9F.

STEP 15: Push PC, GO then RESET. What do you observe?

The familiar rEAdy display indicates the MT-80Z has gone through the initialization process described in Chapter 1.

STEP 16: View the SP. What is the value. How does your observation differ from Step 5?

The Stack Pointer has been set to 1F9F by the RESET function. The program has been interrupted but you will not be able to resume execution from the point of interruption.

EXPERIMENT 8 - INTER KEY

The purpose of this experiment is to help you learn how to use the INTER key. The INTER key is directly wired to the Z80 maskable interrupt $\overline{INT}$ pin. You may recall from the previous experiment that the BREAK key will function any time it is pressed. Before INTER can be used, the interrupt flip-flop (IFF) must be set. This operation unmasks the interrupt input. The IFF is set using the Z80 instruction, EI or by use of the keyboard REG function.

When unmasked, pushing INTER is the equivalent to executing the Z80 instruction, RST 38. The instruction code is FF. The RST 38 calls a subroutine located at Monitor address 0038. This subroutine will cause the Z80 to begin executing instructions starting at an address stored at 1FEE and 1FEF. At power-up, the Monitor stores the address 0066 at 1FEE and 1FEF. This causes the unmasked INTER to function the same as the BREAK key. If you change the contents of 1FEE and 1FEF, the INTER will execute a function <u>you</u> specify.

You will need a short length of #22 or #24 solid hook-up wire for this experiment.

<u>STEP 1</u>: Apply a jumper between PORT SOCKET connections OUT FF and P1 CL. The jumper connects the Port 1 clock input (Port 1 LED interface) to the output address control for port FF. The jumper "maps" the Port 1 Logic Indicators to port address FF. No additional wiring is needed.

STEP 2: Load the program listed below:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	00	NOP	Reserved for later use
1801	97	SUB A	A = 0
1802	D3	OUT (FF),A	Output A to port FF
1803	FF		
1804	0E 06	LD C,FF	C = FF
1805	FF		
1806	10	DJNZ dis	TIME DELAY LOOP
1807	FE		
1808	3C	INC A	A=A+1
1809	18	JR dis	Jump relative to
180A	F7		address 1802

STEP 3: What is the function of this program?

The first instruction, NOP, is a no-operation instruction used to reserve the memory location 1800 for later use. The next two instructions set A to zero and sends a copy of A to the Logic Indicators at port FF. The instructions LD C, FF and DJNZ dis cause a small time delay by decrementing and looping until C = 0. The delay slows the program enabling you to see the LEDs changing. The next two instructions increment the accumulator and create a loop by jumping back to address 1802. The endless loop will continue until interrupted by RESET, BREAK or a properly unmasked INTER.

STEP 4: Run the program. What do you observe at the Port 1 Logic Indicators?

The port 1 LEDs display a very rapid binary count.

STEP 5: Stop the count by pushing BREAK. What do you see?

We observed the address 1806 with instruction Code 10 and the port LEDs showing 1101 0100.

STEP 6: Resume execution by pushing GO.

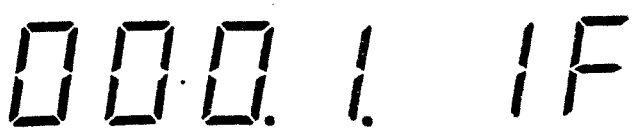
STEP 7: Push INTER. What happens?

Nothing. INTER will not work until the INT input is unmasked.

STEP 8: Push BREAK. Push REG, then I·IF. What do you observe?

The display 0000 1F, indicates the interrupt vector register is zero (left two digits) and the interrupt flip-flop (IFF) is reset (right two digits).

STEP 9: Unmask the INTER function by pushing DATA, then 01. The display should look like this:



STEP 10: Push PC, then GO. The counting on the port 1 LEDs should resume.

STEP 11: Push INTER. What do you observe?

When simply unmasked, INTER functions the same as the BREAK key.

STEP 12: Push GO to resume.

STEP 13: Push INTER. What happens?

When the interrupt from the INTER key was acknowledged by the Z80, further interrupts were disabled. You can verify this by checking the IFF.

STEP 14: Push BREAK, REG, then I·IF. What is the state of the IFF?

The IFF has been reset.

STEP 15: The IFF can be set by the instruction EI (instruction Code FB). Change address 1800 from 00 to FB.

STEP 16: Run the program, then push INTER. What did you observe? Why?

The INTER key operated without manually setting the IFF. This proves the function of the EI instruction.

In the remaining steps, you will change the contents of addresses 1FEE and 1FEF. After this change, INTER will cause the execution of a program you specify.

STEP 17: Enter the following program. Note that the starting address is 1900. The program you entered in Step 1 and changed in Step 15 will remain in memory addresses 1800-180A.

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1900	3E	LD A,55	A = 55
1901	55		
1902	D3	OUT (FF),A	Output A to port FF
1903	FF		
1904	76	HALT	Halt execution of instructions

STEP 18: What is the function of this program?

This program will load A with the number 55, output to the port FF LEDs, then halt.

STEP 19: What are the contents of addresses 1FEE and 1FEF? 1FEE = _____, 1FEF = _____. Why do these addresses contain the values observed?

Using the ADDR function, we found 1FEE = 66 and 1FEF = 00. These values were stored by the Monitor at power-up.

STEP 20: Change 1FEE from 66 to 00. Change 1FEF from 00 to 19.

STEP 21: Push RESET, PC, then GO. You should see the count on the Port 1 LEDs.

STEP 22: Push INTER. What do you observe at the LEDs and HALT indicator? Why?

The port LEDs display 0101 0101 and the HALT LED is lit. This is proof that the program at address 1900 was executed when INTER was pressed.

STEP 23: If you wanted to use INTER to interrupt a program and begin executing instructions at address 18B9, what values must be stored at 1FEE and 1FEF?

1FEE = B9, 1FEF = 18.

STEP 24: How is the INTER key unmasked?

Change the IFF from 00 to 01.

STEP 25: Why does the BREAK key work without unmasking?

BREAK uses the $\overline{\text{NMI}}$, Non-maskable Interrupt input.

4

EXPERIMENT 9 - USER KEY

The purpose of this experiment is to help you learn how to use the USER key. The USER key is the only connection to pin PA6 of the 8255 PPI chip. It is not a function key recognized by the Monitor. The use of this key requires a polling routine which inputs Port A of the 8255 (port 00) and tests bit 6. A conditional instruction can follow the polling routine to branch to a different section of a program depending on the state of bit 6.

Here is an example of a simple polling routine:

```
IN A,(00)      Input PA0-PA7 of 8255
BIT 6,A        Test bit 6 of the accumulator
```

At the end of this routine, the zero flag will be set only if the key was pressed.

STEP 1: Load the following program:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	21	LD HL,1000	HL = 1000
1801	00		
1802	10		
1803	DB	IN A,(00)	A = PA0-PA7 of 8255
1804	00		USER key is PA6
1805	CB	BIT 6,A	Test bit 6 of accumulator
1806	77		
1807	20	JR NZ,dis	If USER not pressed, jump
1808	FA		back to 1803
1809	CD	CALL TONE2K	If USER pressed, call
180A	E2		TONE2K
180B	05		
180C	76	HALT	Stop executing instructions

STEP 2: What is the function of this program?

The first instruction sets HL to 1000 for the TONE2K duration. The next two instructions poll the condition of the USER key. If the key is up, bit 6 = 1 and the Zero flag = 0. If the key is pressed, bit 6 = 0 and the Zero flag = 1. The relative jump instruction, JR NZ, dis, tests the Zero flag. If the key is up, the Zero flag = 0 and the JR NZ, dis causes a jump back to address 1803. This forms a continuous loop until USER is pressed. When the key is pressed, the Zero flag = 1. The

relative jump is skipped and TONE2K is called. A tone from the speaker will last until HL is counted to zero by the TONE2K subroutine. When the subroutine returns, the HALT instruction stops the Z80.

STEP 3: Press PC, then GO. What do you see and hear?

The MT-80Z seems to be "dead" because the Z80 is polling the USER key. It will remain in the polling loop until USER is pressed.

STEP 4: Press USER. What do you observe?

We observed a 2-second tone from the speaker, then the HALT LED lit. Pressing the USER key ends the polling loop allowing execution of the subroutine TONE2k and the HALT instruction.

EXPERIMENT 10 - SPEAKER AND TONE LED

The purpose of this experiment is to demonstrate the direct control of the speaker and the TONE LED. In the experiments, most of the speaker operation is controlled by the Monitor. Occasionally, you borrow the Monitor subroutine, TONE2K as an indicator to verify the operation of a program or function. In this experiment, you will control the speaker and LED directly without the use of Monitor subroutines.

The speaker and TONE LED are both interfaced to the 8255 PC7 pin. The 8255 PC0-PC7 group is mapped as output port 02. See the MT-80Z Technical Reference Manual for the schematic diagram of this circuit.

C A U T I O N

Bit PC6 of port 02 is one of the inputs to the BREAK key interface. Outputting a zero to PC6 is the equivalent to pushing the BREAK key. When using the speaker and TONE LED, it is advised to always maintain bit 6 at a logic 1.

In the next 5 steps, you will learn how to control the green TONE LED.

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	3E	LD A,7F	A = 7F; bit 7 = 0
1802	7F		bits 0-6 = 1
1803	D3	OUT (02),A	8255 PC7 = 0, PC0
1804	02		PC6 = 1, TONE LED is lit
1805	76	HALT	

STEP 2: What is the function of this program?

The first instruction sets the appropriate bit pattern in the accumulator to light the TONE LED. Bit 7 = 0 and the remaining bits, 0-6, are set to 1. It is especially important to have bit 6 = 1 to avoid a BREAK. The second instruction will output the bit pattern, 7F, to output port 02. The TONE LED will light. The HALT instruction will stop program execution.

STEP 3: Push PC, then GO to run the program. What is the state of the TONE and HALT LEDs? Why?

The lighted TONE LED verifies the LD A,7F and OUT (02),A instructions. The HALT LED indicates the HALT state caused by the last instruction of the program.

STEP 4: What is an appropriate bit pattern that would turn off the TONE LED?

We chose FF. Bit 6 must remain set to avoid BREAK.

STEP 5: Change address 1801 from 7F to FF and run the program. What do you observe?

The TONE LED is off and the HALT LED is on.

The speaker is connected to the same interface as the TONE LED. When the LED is on, the speaker voice coil is energized. A program that controls the LED can produce sound from the speaker by the alternation of 1 and 0 at PC7. This is a process called toggling. The following program will toggle at a rapid rate producing a tone from the speaker.

STEP 6: Load the following program:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	97	SUB A	A = 0,CY flag = 0
1801	3E	LD A,FF	A = FF
1802	FF		
1803	D3	OUT (02),A	8255 PC0-PC7 = 1
1804	02		
1805	10	DJNZ dis	DELAY
1806	FE		
1807	1F	RRA	Toggle by rotating CY flag to bit 7
1808	D3	OUT (02),A	8255 PC0-6 = 1,
1809	02		PC7 = 0
180A	17	RLA	Toggle by rotating left, PC7 = 1
180B	18	JR dis	Jump relative to
180C	F4		address 1803

STEP 7: What is the function of the program in Step 6?

The primary purpose of the first instruction is to set the carry flag to zero. The carry flag is used by the rotate instructions in the program. The instructions LD A,FF and OUT (02),A set PC0-PC7 = 1. The voice coil of the speaker is not energized. DNJZ dis causes a short delay by looping to address 1805 until register B = 0. The initial value of B is not important. The RRA instruction rotates the carry flag contents right, into bit 7 of the accumulator. This changes bit 7 from 1 to 0. The next instruction, OUT (02),A sets PC7 to 0. The voice coil is energized. The RLA instruction is opposite to the RRA toggling bit 7 to 1 and the carry flag to 0. JR dis jumps relative back to the OUT (02),A instruction at address 1803. The program loop continually toggles bit 7 of the accumulator and outputs it to PC7 producing the tone.

STEP 8: Push PC, GO, then BREAK. What do you hear? What is the state of the TONE LED? Why?

We heard a raspy tone from the speaker. The TONE LED appears to be out. The program produces a square wave with a very short duty cycle ("on-time") at PC7. The short "on-time" results in a very low level of illumination of the LED.

2

EXPERIMENT 11 - Logic Indicators and the Port Socket

The purpose of this experiment is to help you learn how to use the Logic Indicators to perform the following tasks:

1. Simple logic monitoring

2. Latching and displaying bus data sent from the Z80 using the series of OUT instructions.

The use of the Port 1 Logic Indicators as a data bus monitor will be covered in Experiment 13. Before proceeding with this experiment you are advised to review the Chapter 1 sections describing both the Logic Indicators and the PORT SOCKET.

For this experiment, the following jumpers are required.

8 each 6" (15.2 cm)
3 each 3" (7.6 cm)

Use #22 or #24 solid hook-up wire. Strip each end of the jumpers approximately 3/8" (1 cm).

In the next 5 steps, you will learn how to use the Port 2 LEDs as simple logic monitors.

<u>STEP 1</u>: Connect the end of a 6" jumper to PORT SOCKET connector L7. You can use the other end of the jumper as a logic probe. Connect the jumper to +5 on the PORT SOCKET. What do you observe on the Port 2 LED 7? Why?

+5V is a logic 1 state indicated by the light at LED 7.

STEP 2: Connect the jumper to GND on the PORT SOCKET. What do you see?

LED 7 is out indicating a logic 0 state.
Note: When L7 is not connected, the LED is also out.

STEP 3: All the LEDs of Port 2 can be used as a logic probe. Connect one end of the jumper to +5. Move the other end from L7 to L6 then L5 and so on until you have tested the response of all eight Port 2 LEDs. What do you observe?

Any of the Port 2 LEDs light when connected to logic 1 state.

STEP 4: Let's monitor a logic level on the Z80 bus. Connect the jumper from L0 to $\overline{\text{SY RST}}$ on the BUS SOCKET. What is indicated on the LED?

The normal condition of the bus connection is logic 1. What will cause it to go to logic 0?

A RESET input.

STEP 5: Push the RESET key a few times. What do you observe?

THE LED goes out each time the RESET key is pushed. A logic 0 indicates a reset input.

In the remaining steps of this experiment, you will learn how to use the Port 1 and Port 2 Logic Indicators as output ports. The term, output port, refers to the LEDs and chips that are in contact with the Z80 data bus whenever an output instruction is executed.

The PORT SOCKET provides port addresses FD, FE and FF. These addresses can be assigned to Port 1 and Port 2. Port 2 can be split into Port 2X (4-bit) and Port 2Y (4-bit). Port 1 is permanently interfaced to the data bus. Port 2 use requires jumpering to the data bus on the BUS SOCKET.

In the next 7 steps, you will learn how to use the port 1 LEDs as output port FD, FE or FF. NOTE: Connect a jumper between P1CL and OUTFF.

STEP 7: Load the following program:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	3E	LD A, 55	A = 55
1801	55		
1802	D3	OUT (FF),A	Output register A to
1803	FF		port FF
1804	76	HALT	Stop executing instructions

STEP 8: What is the function of the program in Step 7?

The first instruction sets the accumulator to the value 55. The second instruction, OUT (FF),A sends a copy of the accumulator to port FF. The HALT instruction stops the Z80 and lights the HALT LED.

STEP 9: Push PC, then GO. What do you observe?

The LEDs at Port 1 indicate a binary 01010101 or hexadecimal 55. This proves the operation of the program and the mapping of the Port 1 LEDs to port address FF. How would you change the program to display the following bit pattern at Port 1: 10101010?

Change the code at address 1801 to AA. Try it. Push RESET to exit the HALT condition. Push PC, then NEXT to display address 1801. Enter AA to change address 1801. Push PC, then GO to execute the program. Did it work?

We were able to display any bit pattern from 00 to FF.

STEP 10: Port 1 can also be mapped to ports FD and FE Move the jumper from OUT FF to OUT FE. The Port 1 LEDs are now mapped to port FE. This change in hardware (moving the jumper) requires a modification of the software. Which instruction must be changed?

Change OUT (FF),A to OUT (FE),A. The code at address 1803 must be changed to FE.

.06

STEP 11: Change 1803 from FF to FE and run the program. What are the results? Why?

Port 1 displays the accumulator contents because the Z80 output instruction and the port hardware match. What changes in jumpering is required to use the instruction OUT (FD),A?

Move the jumper from OUT FE to OUT FD.

STEP 12: Change the jumper and instruction. Run the program. What do you observe?

As long as the jumper (hardware) matches the OUT (PORT),A instruction (software) the port LEDs will display accumulator contents. The Port 1 LEDs can be mapped to ports FD, FE or FF.

In the next 5 steps, you will learn how to use the Port 2 Logic Indicators as output ports FD, FE or FF.

STEP 13: Remove power and use jumpers to make the following connections:

a. a short jumper from OUT FF to P2X CL
b. a short jumper from P2X CL to P2Y CL.

NOTE: The PORT SOCKET provides five electrically connected terminals for each labeled function. They are arranged in a vertical row above the label. The second wire connected to P2X CL can be inserted above or below the existing wire.

c. Eight long jumpers from L0-L7 of the PORT SOCKET to D0-D7 of the BUS SOCKET. Be sure that the numbers correspond: L0 to D0, L1 to D1, etc.

STEP 14: Check your wiring, then apply power. The "rEAdy" display should appear on the 7-segment display. If the display appears blank, remove power and recheck the jumpers. The most likely problem is an incorrect jumper to the data bus.

STEP 15: Load the program listed in Step 7.

STEP 16: Run the program. What do you observe on the Port 2 LEDs?

The LEDs at Port 2 indicate a binary 01010101 or hexadecimal 55. This proves the operation of the program and the mapping of the Port 2 LEDs to port FF.

STEP 17: The Port 2 Logic Indicators can also be mapped to ports FD and FE. Move the jumper from OUT FF to OUT FE and repeat Steps 10, 11 and 12.

How is Port 2 similar to Port 1?

Both Port 1 and Port 2 Logic Indicators can be mapped to port addresses FD, FE and FF. How are Port 1 and Port 2 different?

Port 1 is permanently interfaced to the data bus and requires only a single jumper for port address. Port 2 requires jumpers to the data bus and 2 jumpers for port address.

In the remaining steps, you will learn how to use the Port 2 Logic Indicators as two independent 4-bit output ports.

STEP 18: Remove the 2 jumpers used for port address control. The 8 jumpers from L0-L7 to D0-D7 should remain in place.

STEP 19: Use two short jumpers and make the following PORT SOCKET connections:

a. P2X CL to OUT FE
b. P2Y CL to OUT FD

Port 2 is now split into 4-bit output ports. What is the port address of Port 2X and Port 2Y?

Port 2X is mapped as port FE and Port 2Y is mapped as port FD. Port 2X uses LEDs 0-3 and Port 2Y uses LEDs 4-7.

STEP 20: How can the program listed in Step 7 be changed to cause the following bit pattern to be displayed?

Port 2Y	Port 2X
1100	0011

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	3E	LD A,03	A = 03
1801	03		
1802	D3	OUT (FE),A	Output A to port FE
1803	FE		
1804	3E	LD A,C0	A = C0
1805	C0		
1806	D3	OUT (FD),A	Output A to port FD
1807	FD		
1809	76	HALT	Stop executing instructions

STEP 21: What is the function of the program listed in Step 20?

The first instruction loads A with 03. The OUT (FE),A sends 03 to port FE. However, port FE is connected only to data bus D0-D3. Port FE latches and displayes the 3. The next instruction loads A with C0. The OUT (FD),A sends a copy of the entire accumulator contents, C0, to port FD. Port FD is interfaced to D4-D7. and displays only the C (1100).

STEP 22: Load and STEP the program listed in Step 20. What is the bit pattern displayed on the Port 2 LEDs?

1100 0011

STEP 23: How would you map Port 2X to port FF?

Move the jumper from OUT FE to OUT FF.

EXPERIMENT 12 - LOGIC SWITCHES AND THE PORT SOCKET

The purpose of this experiment is to help you learn how to use the Logic Switches to perform the following tasks:

1. Use Port 2 to apply a logic 1 or 0 to PORT SOCKET terminals S0-S7. These terminals can provide an external stimulus to the inputs of interfacing circuits assembled on the breadboarding socket.

2. Use Ports 1 and 2 to supply data to the Z80 as input ports.

Before proceeding with this experiment, you are advised to review the Chapter 1 sections describing both the Logic Indicators and the PORT SOCKET.

For this experiment, you will require the following jumpers:

8 each 6" (15.2 cm)
3 each 3" (7.6 cm)

Use #22 or #24 solid hook-up wire. Strip each end of the jumpers approximately 3/8" (1 cm).

In the next 5 steps, you will learn how to use the Port 2 Logic Switches to apply a logic 1 or 0 to a digital circuit.

<u>STEP 1</u>: Connect a 6" jumper from PORT SOCKET S0 to L0. Now, use 7 jumpers to connect the remaining terminals S1-S7 to L1-L7. The numbers should match.

STEP 2: Change some of the Port 2 switches. What do you observe on the Port 2 LEDs? Why?

We observed a bit pattern on the LEDs that match the switch settings. The Port 2 LEDs are used as logic monitors indicating a light for a logic 1 and no light for a logic 0.

STEP 3: How is the switch rocker positioned for a logic 1?

The rocker is pushed toward the word OPEN printed on the switch to select a logic 1. Set the switches to observe a binary 1111 1111 on the Port 2 LEDs.

STEP 4: There are two sets of numbers displayed near the switches. The switches themselves are numbered 1-8. The board is marked 0-7. Which set of numbers correspond to the PORT SOCKET numbering for S0-S7?

0-7.

STEP 5: How is the rocker positioned for a logic 0?

The rocker is pushed toward the numbers to select a logic 0. Set the switches to observe a binary 0000 0000.

In the next 5 steps, you will learn how Port 2 can be split into Port 2X and Port 2Y. Also, you will learn how P2X EN and P2Y EN provide tristate control for PORT SOCKET terminals S0-S7 (switches).

.14

STEP 6: Using a short jumper, connect P2X EN to P2Y EN.

STEP 7: Set the Port 2 switches to display 1111 1111 on the logic monitor LEDs.

NOTE: The PORT SOCKET provides five electrically connected terminals for each labeled function. They are arranged in a vertical row above the label. In the next step you will be required to make a second connection to P2Y EN. The jumper wire can be inserted above or below the existing wire.

STEP 8: Using a short jumper, connect P2Y EN to IN FD. What do you see?

The logic monitor LEDs display 0000 0000. Is the logic level at S0-S7 a logic 0?

No. The connection of P2Y EN and P2X EN to the input port control IN FD caused a high impedance or "tristate" condition at outputs S0-S7. The connection of an input port to the data bus of a microcomputer requires tristate control. The bidirectional data bus concept allows only one transmitting unit at a time. Tristate controls act as the traffic signal to input ports and memory systems to avoid conflicts of information being transmitted (bus contention).

STEP 9: Remove the jumper connecting P2Y EN to P2X EN. What do you observe? Why?

Our switches were set to 11111111 and we observed 0000 1111 on the Port 2 LEDs. Without the jumper, one-half of the Port 2 Logic Switches are enabled ("de-tristated"). Port 2 can be split into two ports: Port 2X and Port 2Y. Which half of Port 2 is enabled? Which half is in the tristate (high Z) condition?

Port 2X, S0-S3, is enabled. Port 2Y, S4-S7, is tristated.

STEP 10: Move the jumper from P2Y EN to P2X EN. What do you observe?

P2X is tristated and P2Y is enabled.

In the next 11 steps, you will learn how to use the Port 2 Logic Switches as an input port. The port addresses available for Port 2 are FD, and FE.

STEP 11: Remove power and make the following connections on the PORT and BUS sockets.

a. P2X EN to P2Y EN
b. P2Y EN to IN FD
c. P1 CL to OUT FF
d. S0-S7 on the PORT SOCKET to D0-D7

The port 2 switches are now interfaced to the MT-80Z data bus as input port FD. The Port 1 Logic Indicators are interfaced to the data bus as output port FF.

STEP 12: Apply power. You should see the rEAdy display. If the 7-segment display is blank, you have misplaced the jumpers for connections a and b of Step 11. Look for shorts between jumpers connected to the data bus.

STEP 13: Load the following program:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	DB	IN A,(FD)	Input switch data to
1801	FD		accumulator
1802	D3	OUT (FF),A	Output accumulator to
1803	FF		port FF
1804	18	JR dis	Jump relative to
1805	FA		address 1800

STEP 14: What is the function of the program listed in Step 13?

The first instruction, IN A,(FD) inputs the switch states, S0-S7, to the accumulator. The second instruction, OUT (FF),A outputs a copy of the accumulator to the port FF LEDs. The last instruction forms a continuous loop by jumping back to address 1800.

STEP 15: Push PC, then GO. Set the Port 2 Logic switches to 0000 1111. What do you observe on the Port 1 LEDs? Try some other switch combinations.

We observed 0000 1111 on the LEDs. When any of the switches are changed to logic 1, the output port LED would light.

STEP 16. Very carefully remove the jumper to disconnect D7 from S7. What do you observe?

The output port Led 7 remains lit regardles of switch settings. Data bus connection D7 is "floating" high during the IN A,(FD) instruction. Whenever an input instruction is executed, the ENTIRE 8-bit data bus is stored in the Z80 accumulator register.

STEP 17: Remove power from the MT-80Z and make the following connections:

a. Replace the jumper from S7 to D7. You should have S0-S7 connected to D0-D7.
b. P2X EN to IN FE
c. P2Y EN to IN FD
d. P1 CL to OUT FF.

STEP 18: Apply power. You should see the rEAdy display. If the 7-segment display is blank, recheck your connections.

STEP 19: Load and run the program listed in Step 13.

STEP 20: Change the settings of Port 2 switches 4,5,6 and 7. Do you see any changes at the Port 1 LEDs? Try switches 0,1,2 and 3. Does it cause the same response at Port 1? Can you explain your observation?

We observed switches 4-7 lighting Port 1 LEDs 4-7. However, LEDs 0-3 appear lit regardless of switches 0-3. The program inputs port FD which consists of P2Y switches S4-S7. The remaining switches, P2X are mapped as input port FE. This port is not selected by the program and remains in the tristate condition at all times. Why do LEDs 0-3 remain lit?

The "tristate" condition is a high impedance state that allows the data bus (D0-D3) to "float high" during execution of the input instruction. The LEDs prove the floating condition by indicating a constant logic 1 state.

STEP 21: What would happen if the instruction, IN A,(FD) was changed to IN A,(FE)? Change the byte at 1801 from FD to FE and run the program.

We observed Port 1 LEDs 4-7 lit indicating the data bus connections D4-D7 were floating. We were able to control LEDs 0-3 by operating Port 2 switches 0-3.

STEP 22: Remove power from the MT-80Z and remove all jumpers.

In the remaining steps of this experiment, you will learn how to use the Port 1 Logic switches as input port FF. The Port 1 switches are permanently interfaced to the data bus. Using the PORT SOCKET, Port 1 can be mapped only to input port FF.

NOTE: Both $\overline{\text{P1 EN}}$ and $\overline{\text{IN FF}}$ are active low.

STEP 23: Use two short jumpers and make the following PORT SOCKET connections:

a. P1 CL to OUT FF
b. $\overline{\text{P1 EN}}$ to $\overline{\text{IN FF}}$

STEP 24: Apply power and load the program listed in Step 13 and change: IN A,(FD) to IN A,(FF). This requires changing the code at address 1801 from FD to FF.

STEP 25: Run the program and change the state of the Port 1 switches. What do you see?

We observed the Port 1 LEDs being controlled by the Port 1 Logic Switches.

EXPERIMENT 13 - SINGLE CYCLE OPERATION

The purpose of this experiment is to help you learn how to use the SINGLE CYCLE feature and the bus monitor to observe the effects of each Z80 machine cycle during the execution of a program. The switches used for this experiment are S3, the CYCLE-RUN switch and PB1, the single cycle step switch. If there are no connections to P1 CL on the PORT SOCKET, the Port 1 Logic Indicators are used as a Z80 data bus monitor.

In this experiment, you will execute a short program using SINGLE CYCLE stepping and observe the data bus changes. These observations provide an in-depth view of computer operations.

STEP 1: Connect a short jumper between PORT SOCKET connections CYCLE and WAIT. Use #22 or #24 solid hook-up wire.

STEP 2: Set the CYCLE-RUN switch to RUN.

STEP 3: Load the following program:

ADDRESS	INSTRUCTION CODE	MNEMONIC	COMMENTS
1800	21	LD HL,1810	HL = 1810
1801	10		
1802	18		
1803	34	INC (HL)	(1810) = (1810) +1
1804	18	JR dis	Jump relative back
1805	FA		to 1800

STEP 4: What is the function of this program?

The first instruction sets register pair HL to 1810. HL will be used as an address pointer for the next instruction. INC (HL) will increment the contents of a memory location addressed by HL. In this case, the contents of address 1810 will be incremented. The last instruction forms a continuous loop by jumping back to address 1800. When running at full speed, this program continuously increments the contents of address 1810. Because the program is a loop, the initial contents of 1810 is not important.

The use of SINGLE CYCLE stepping will allow you to monitor the data bus for every machine cycle of the program. It is particularly interesting to view the bus during the cycles of the INC (HL) instruction.

STEP 5: With CYCLE-RUN in the RUN position, push PC, then GO. The blank 7-segment display indicates the program loop is in execution.

STEP 6: Switch CYCLE-RUN to CYCLE. What is displayed in the Port 1 bus monitor?

We observed the hex value 18. Switching to CYCLE stopped the program at a random location in the loop. Your display is very likely to be different. The present task is to determine where you are within the program loop.

STEP 7: Push the cycle step button, PB1, once. What do you see?

Our bus monitor displayed 34. This is the code for the instruction INC (HL) being fetched by the Z80.

STEP 8: Press PB1 six more times, noting the data bus monitor display (Port 1 LEDs).

STEP 9: Use the following listing of single cycle bus displays to determine your location in the program loop:

```
21        LD   HL, 1810
10
18
34        INC  (HL)
--
--
18        JR dis
FA
```

STEP 10: Press PB1 until the bus monitor displays the instruction code for INC (HL): 34.

STEP 11: Press PB1 once and note the display.

STEP 12: Press PB1 again and note the display.

For steps 11 and 12, we observed 85 and 86. Why are the values consecutive numbers?

The execution of INC (HL) requires the following three machine cycles:

Bus contents	Cycle description
34	Instruction fetch
--	Read contents of address 1810
--	Write incremented contents of 1810

STEP 13: How many machine cycles are required to execute LD HL,1810?

Three.
How many machine cycles are required to execute JR dis?

Two.
How many presses of PB1 are required to execute the entire program loop?

Eight.

STEP 14: How can you stop single cycle stepping and view or modify registers, flags and memory contents?

The BREAK key can be used before switching from CYCLE to RUN.

STEP 15: Push PB1 until the bus monitor displays the contents of address 1810. This number will be displayed immediately following the instruction Code, 34.

STEP 16: Push BREAK. There will not be any change in the LED displays.

STEP 17: Switch CYCLE-RUN to RUN. What do you observe on the 7-segment display?

When you switch to run after BREAK, the Z80 completes the instruction, INC (HL), then acknowledges the BREAK interrupt. The 7-segment display should show the next instruction code and address: 1804 1.8.

STEP 18: Use the ADDR function to examine the contents of address 1810. What do you find?

When we did this step, the contents of 1810 were one more than the value displayed on the bus monitor in Step 15. The INC (HL) instruction was completed before the BREAK input gained control of the MT-80Z.

APPENDIX 1

BINARY	HEXADECIMAL	DECIMAL
0000	00	0
0001	01	1
0010	02	2
0011	03	3
0100	04	4
0101	05	5
0110	06	6
0111	07	7
1000	08	8
1001	09	9
1010	0A	10
1011	0B	11
1100	0C	12
1101	0D	13
1110	0E	14
1111	0F	15
10000	10	16

APPENDIX 2

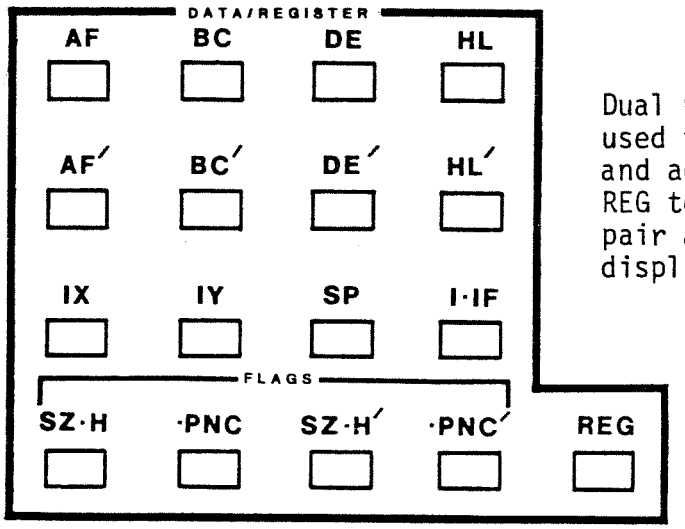
MT-80Z Keyboard Quick Reference List

ADDR	Push ADDR, then number keys to enter addresses.	Page 1-23
BREAK	Break program execution, save registers and flags, display break address and data. Interfaced to Z80 $\overline{NMI}$.	Page 1-21
CLR BRK PT	Clear breakpoint, display shows F.F.F.F. F.F.	Page 1-24
SET BRK PT	Set breakpoint at displayed address. MT-80Z allows one breakpoint. Setting new breakpoint automatically clears previous breakpoint.	Page 1-24
COPY	Transfers a copy of a block of memory to another area of memory. Format: Push COPY, enter starting address of block, push NEXT, enter ending address of block, push NEXT, enter destination starting address, push GO.	Page 1-24
DATA	Push DATA, then number keys to enter memory, register or flag contents.	Page 1-23
DELETE	Delete the displayed memory contents and move subsequent memory contents to the next low address.	Page 1-24
DUMP	Store MT-80Z memory contents on audio tape recorder. Format: Push DUMP, enter hexadecimal file name, push NEXT, enter starting address of data to be recorded, push NEXT, enter ending address of data to be recorded, push GO.	Page 1-25
GO	Run program starting at displayed address. Format: Press PC (or set address using ADDR), then GO.	Page 1-25

INSERT	Insert one byte into memory at the address displayed +1. Moves subsequent memory contents to the next highest address.	Page 1-24
INTER	Maskable interrupt. Uses vector that Monitor loads at 1FEE and 1FEF. INTER is equivalent to RST 38 instruction. Connected to Z80 $\overline{\text{INT}}$.	Page 1-22
LOAD	Load data recorded by MT-80Z DUMP command from audio tape recorder to MT-80Z memory. Format: Push LOAD, enter hexadecimal file name, push GO.	Page 1-25
NEXT	Increment displayed address, move to next register pair or flag group display.	Page 1-23
PC	Display user's program counter. User's PC can be changed prior to pressing GO or STEP.	Page 1-25
PREV	Decrement displayed address, move to previous register pair or flag group display.	Page 1-23
RELA	Calculate and store displacement for JR and DJNZ instructions. Format: Push RELA, enter address of JR or DJNZ instruction, push NEXT, enter destination of jump, push GO.	Page 1-23
RESET	System reset, interfaced to Z80 $\overline{\text{RST}}$.	Page 1-21
STEP	Single instruction step key. Format: Press PC (or set address using ADDR), then STEP each instruction.	Page 1-26

JSER
KEY

Connecter to PA6 of 8255, mapped as input Port 00. Page 1-26



Dual purpose keys, used to input data and addresses. Push REG to select register pair and flag group displays. Page 1-20

Z80–CPU INSTRUCTIONS SORTED BY MNEMONIC

OBJ CODE	SOURCE STATEMENT
8E	ADC A,(HL)
DD8E05	ADC A,(IX+d)
FD8E05	ADC A,(IY+d)
8F	ADC A,A
88	ADC A,B
89	ADC A,C
8A	ADC A,D
8B	ADC A,E
8C	ADC A,H
8D	ADC A,L
CE20	ADC A,N
ED4A	ADC HL,BC
ED5A	ADC HL,DE
ED6A	ADC HL,HL
ED7A	ADC HL,SP
86	ADD A,(HL)
DD8605	ADD A,(IX+d)
FD8605	ADD A,(IY+d)
87	ADD A,A
80	ADD A,B
81	ADD A,C
82	ADD A,D
83	ADD A,E
84	ADD A,H
85	ADD A,L
C620	ADD A,N
09	ADD HL,BC
19	ADD HL,DE
29	ADD HL,HL
39	ADD HL,SP
DD09	ADD IX,BC
DD19	ADD IX,DE
DD29	ADD IX,IX
DD39	ADD IX,SP
FD09	ADD IY,BC
FD19	ADD IY,DE
FD29	ADD IY,IY
FD39	ADD IY,SP
A6	AND (HL)
DDA605	AND (IX+d)
FDA605	AND (IY+d)
A7	AND A
A0	AND B
A1	AND C
A2	AND D
A3	AND E
A4	AND H
A5	AND L
E620	AND N
CB46	BIT 0,(HL)
DDCB0546	BIT 0,(IX+d)
FDCB0546	BIT 0,(IY+d)
CB47	BIT 0,A
CB40	BIT 0,B
CB41	BIT 0,C
CB42	BIT 0,D
CB43	BIT 0,E
CB44	BIT 0,H
CB45	BIT 0,L
CB4E	BIT 1,(HL)
DDCB054E	BIT 1,(IX+d)
FDCB054E	BIT 1,(IY+d)
CB4F	BIT 1,A
BC48	BIT 1,B
CB49	BIT 1,C
CB4A	BIT 1,D
CB4B	BIT 1,E
CB4C	BIT 1,H
CB4D	BIT 1,L
CB56	BIT 2,(HL)
DDCB0556	BIT 2,(IX+d)
FDCB0556	BIT 2,(IY+d)
CB57	BIT 2,A
CB50	BIT 2,B
CB51	BIT 2,C
CB52	BIT 2,D
CB53	BIT 2,E
CB54	BIT 2,H
CB55	BIT 2,L
CB5E	BIT 3,(HL)
DDCB055E	BIT 3,(IX+d)
FDCB055E	BIT 3,(IY+d)
CB5F	BIT 3,A
CB58	BIT 3,B
CB59	BIT 3,C
CB5A	BIT 3,D
CB5B	BIT 3,E
CB5C	BIT 3,H
CB5D	BIT 3,L
CB66	BIT 4,(HL)
DDCB0566	BIT 4,(IX+d)
FDCB0566	BIT 4,(IY+d)
CB67	BIT 4,A
CB60	BIT 4,B
CB61	BIT 4,C
CB62	BIT 4,D
CB63	BIT 4,E
CB64	BIT 4,H
CB65	BIT 4,L
CB6E	BIT 5,(HL)
DDCB056E	BIT 5,(IX+d)
FDCB056E	BIT 5,(IY+d)
CB6F	BIT 5,A
CB68	BIT 5,B
CB69	BIT 5,C
CB6A	BIT 5,D
CB6B	BIT 5,E
CB6C	BIT 5,H
CB6D	BIT 5,L
CB76	BIT 6,(HL)
DDCB0576	BIT 6,(IX+d)
FDCB0576	BIT 6,(IY+d)
CB77	BIT 6,A
CB70	BIT 6,B
CB71	BIT 6,C
CB72	BIT 6,D
CB73	BIT 6,E
CB74	BIT 6,H
CB75	BIT 6,L
CB7E	BIT 7,(HL)
DDCB057E	BIT 7,(IX+d)
FDCB057E	BIT 7,(IY+d)
CB7F	BIT 7,A
CB78	BIT 7,B
CB79	BIT 7,C
CB7A	BIT 7,D
CB7B	BIT 7,E
CB7C	BIT 7,H
CB7D	BIT 7,L
DC8405	CALL C,NN
FC8405	CALL M,NN
D48405	CALL NC,NN
CD8405	CALL NN
C48405	CALL NZ,NN
F48405	CALL P,NN
EC8405	CALL PE,NN
E48405	CALL PO,NN
CC8405	CALL Z,NN
3F	CCF
BE	CP (HL)
DDBE05	CP (IX+d)
FDBE05	CP (IY+d)
BF	CP A
B8	CP B
B9	CP C
BA	CP D
BB	CP E
BC	CP H

CB99	RES 3,C
CB9A	RES 3,D
CB9B	RES 3,E
CB9C	RES 3,H
CB9D	RES 3,L
CBA6	RES 4,(HL)
DDCB05A6	RES 4,(IX+d)
FDCB05A6	RES 4,(IY+d)
CBA7	RES 4,A
CBA0	RES 4,B
CBA1	RES 4,C
CBA2	RES 4,D
CBA3	RES 4,E
CBA4	RES 4,H
CBA5	RES 4,L
CBAE	RES 5,(HL)
DDCB05AE	RES 5,(IX+d)
FDCB05AE	RES 5,(IY+d)
CBAF	RES 5,A
CBA8	RES 5,B
CBA9	RES 5,C
CBAA	RES 5,D
CBAB	RES 5,E
CBAC	RES 5,H
CBAD	RES 5,L
CBB6	RES 6,(HL)
DDCB05B6	RES 6,(IX+d)
FDCB05B6	RES 6,(IY+d)
CBB7	RES 6,A
CBB0	RES 6,B
CBB1	RES 6,C
CBB2	RES 6,D
CBB3	RES 6,E
CBB4	RES 6,H
CBB5	RES 6,L
CBBE	RES 7,(HL)
DDCB05BE	RES 7,(IX+d)
FDCB05BE	RES 7,(IY+d)
CBBF	RES 7,A
CBB8	RES 7,B
CBB9	RES 7,C
CBBA	RES 7,D
CBBB	RES 7,E
CBBC	RES 7,H
CBBD	RES 7,L
C9	RET
D8	RET C
F8	RET M

D0	RET NC
C0	RET NZ
F0	RET P
E8	RET PE
E0	RET PO
C8	RET Z
ED4D	RETI
ED45	RETN
CB16	RL (HL)
DDCB0516	RL (IX+d)
FDCB0516	RL (IY+d)
CB17	RL A
CB10	RL B
CB11	RL C
CB12	RL D
CB13	RL E
CB14	RL H
CB15	RL L
17	RLA
CB06	RLC (HL)
DDCB0506	RLC (IX+d)
FDCB0506	RLC (IY+d)
CB07	RLC A
CB00	RLC B
CB01	RLC C
CB02	RLC D
CB03	RLC E
CB04	RLC H
CB05	RLC L
07	RLCA
ED6F	RLD
CB1E	RR (HL)
DDCB051E	RR (IX+d)
FDCB051E	RR (IY+d)
CB1F	RR A
CB18	RR B
CB19	RR C
CB1A	RR D
CB1B	RR E
CB1C	RR H
CB1D	RR L
1F	RRA
CB0E	RRC (HL)
DDCB050E	RRC (IX+d)
FDCB050E	RRC (IY+d)
CB0F	RRC A
CB08	RRC B
CB09	RRC C

CB0A	RRC D
CB0B	RRC E
CB0C	RRC H
CB0D	RRC L
0F	RRCA
ED67	RRD
C7	RST 0
D7	RST 10H
DF	RST 18H
E7	RST 20H
EF	RST 28H
F7	RST 30H
FF	RST 38H
CF	RST 8
9E	SBC A,(HL)
DD9E05	SBC A,(IX+d)
FD9E05	SBC A,(IY+d)
9F	SBC A,A
98	SBC A,B
99	SBC A,C
9A	SBC A,D
9B	SBC A,E
9C	SBC A,H
9D	SBC A,L
DE20	SBC A,N
ED42	SBC HL,BC
ED52	SBC HL,DE
ED62	SBC HL,HL
ED72	SBC HL,SP
37	SCF
CBC6	SET 0,(HL)
DDCB05C6	SET 0,(IX+d)
FDCB05C6	SET 0,(IY+d)
CBC7	SET 0,A
CBC0	SET 0,B
CBC1	SET 0,C
CBC2	SET 0,D
CBC3	SET 0,E
CBC4	SET 0,H
CBC5	SET 0,L
CBCE	SET 1,(HL)
DDCB05CE	SET 1,(IX+d)
FDCB05CE	SET 1,(IY+d)
CBCF	SET 1,A
CBC8	SET 1,B
CBC9	SET 1,C
CBCA	SET 1,D
CBCB	SET 1,E

DD4E05	LD C,(IX+d)
FD4E05	LD C,(IY+d)
4F	LD C,A
48	LD C,B
49	LD C,C
4A	LD C,D
4B	LD C,E
4C	LD C,H
4D	LD C,L
0E20	LD C,N
56	LD D,(HL)
DD5605	LD D,(IX+d)
FD5605	LD D,(IY+d)
57	LD D,A
50	LD D,B
51	LD D,C
52	LD D,D
53	LD D,E
54	LD D,H
55	LD D,L
1620	LD D,N
ED5B8405	LD DE,(NN)
118405	LD DE,NN
5E	LD E,(HL)
DD5E05	LD E,(IX+d)
FD5E05	LD E,(IY+d)
5F	LD E,A
58	LD E,B
59	LD E,C
5A	LD E,D
5B	LD E,E
5C	LD E,H
5D	LD E,L
1E20	LD E,N
66	LD H,(HL)
DD6605	LD H,(IX+d)
FD6605	LD H,(IY+d)
67	LD H,A
60	LD H,B
61	LD H,C
62	LD H,D
63	LD H,E
64	LD H,H
65	LD H,L
2620	LD H,N
2A8405	LD HL,(NN)
218405	LD HL,NN
ED47	LD I,A

DD2A8405	LD IX,(NN)
DD218405	LD IX,NN
FD2A8405	LD IY,(NN)
FD218405	LD IY,NN
6E	LD L,(HL)
DD6E05	LD L,(IX+d)
FD6E05	LD L,(IY+d)
6F	LD L,A
68	LD L,B
69	LD L,C
6A	LD L,D
6B	LD L,E
6C	LD L,H
6D	LD L,L
2E20	LD L,N
ED7B8405	LD SP,(NN)
F9	LD SP,HL
DDF9	LD SP,IX
FDF9	LD SP,IY
318405	LD SP,NN
EDA8	LDD
EDB8	LDDR
EDA0	LDI
EDB0	LDIR
ED44	NEG
00	NOP
B6	OR (HL)
DDB605	OR (IX+d)
FDB605	OR (IY+d)
B7	OR A
B0	OR B
B1	OR C
B2	OR D
B3	OR E
B4	OR H
B5	OR L
F620	OR N
EDBB	OTDR
ED83	OTIR
ED79	OUT (C),A
ED41	OUT (C),B
ED49	OUT (C),C
ED51	OUT (C),D
ED59	OUT (C),E
ED61	OUT (C),H
ED69	OUT (C),L
D320	OUT (N),A
EDAB	OUTD

EDA3	OUTI
F1	POP AF
C1	POP BC
D1	POP DE
E1	POP HL
DDE1	POP IX
FDE1	POP IY
F5	PUSH AF
C5	PUSH BC
D5	PUSH DE
E5	PUSH HL
DDE5	PUSH IX
FDE5	PUSH IY
CB86	RES 0,(HL)
DDCB0586	RES 0,(IX+d)
FDCB0586	RES 0,(IY+d)
CB87	RES 0,A
CB80	RES 0,B
CB81	RES 0,C
CB82	RES 0,D
CB83	RES 0,E
CB84	RES 0,H
CB85	RES 0,L
CB8E	RES 1,(HL)
DDCB058E	RES 1,(IX+d)
FDCB058E	RES 1,(IY+d)
CB8F	RES 1,A
CB88	RES 1,B
CB89	RES 1,C
CB8A	RES 1,D
CB8B	RES 1,E
CB8C	RES 1,H
CB8D	RES 1,L
CB96	RES 2,(HL)
DDCB0596	RES 2,(IX+d)
FDCB0596	RES 2,(IY+d)
CB97	RES 2,A
CB90	RES 2,B
CB91	RES 2,C
CB92	RES 2,D
CB93	RES 2,E
CB94	RES 2,H
CB95	RES 2,L
CB9E	RES 3,(HL)
DDCB059E	RES 3,(IX+d)
FDCB059E	RES 3,(IY+d)
CB9F	RES 3,A
CB98	RES 3,B

BD	CP L
FE20	CP N
EDA9	CPD
EDB9	CPDR
EDA1	CPI
EDB1	CPIR
2F	CPL
27	DAA
35	DEC (HL)
DD3505	DEC (IX+d)
FD3505	DEC (IY+d)
3D	DEC A
05	DEC B
0B	DEC BC
0D	DEC C
15	DEC D
1B	DEC DE
1D	DEC E
25	DEC H
2B	DEC HL
DD2B	DEC IX
FD2B	DEC IY
2D	DEC L
3B	DEC SP
F3	DI
102E	DJNZ DIS
FB	EI
E3	EX (SP),HL
DDE3	EX (SP),IX
FDE3	EX (SP),IY
08	EX AF,AF'
EB	EX DE,HL
D9	EXX
76	HALT
ED46	IM 0
ED56	IM 1
ED5E	IM 2
ED78	IN A,(C)
DB20	IN A,(N)
ED40	IN B,(C)
ED48	IN C,(C)
ED50	IN D,(C)
ED58	IN E,(C)
ED60	IN H,(C)
ED68	IN L,(C)
34	INC (HL)
DD3405	INC (IX+d)
FD3405	INC (IY+d)

3C	INC A
04	INC B
03	INC BC
0C	INC C
14	INC D
13	INC DE
1C	INC E
24	INC H
23	INC HL
DD23	INC IX
FD23	INC IY
2C	INC L
33	INC SP
EDAA	IND
EDBA	INDR
EDA2	INI
EDB2	INIR
E9	JP (HL)
DDE9	JP (IX)
FDE9	JP (IY)
DA8405	JP C,NN
FA8405	JP M,NN
D28405	JP NC,NN
C38405	JP NN
C28405	JP NZ,NN
F28405	JP P,NN
EA8405	JP PE,NN
E28405	JP PO,NN
CA8405	JP Z,NN
382E	JR C,DIS
182E	JR DIS
302E	JR NC,DIS
202E	JR NZ,DIS
282E	JR Z,DIS
02	LD (BC),A
12	LD (DE),A
77	LD (HL),A
70	LD (HL),B
71	LD (HL),C
72	LD (HL),D
73	LD (HL),E
74	LD (HL),H
75	LD (HL),L
3620	LD (HL),N
DD7705	LD (IX+d),A
DD7005	LD (IX+d),B
DD7105	LD (IX+d),C
DD7205	LD (IX+d),D

DD7305	LD (IX+d),E
DD7405	LD (IX+d),H
DD7505	LD (IX+d),L
DD360520	LD (IX+d),N
FD7705	LD (IY+d),A
FD7005	LD (IY+d),B
FD7105	LD (IY+d),C
FD7205	LD (IY+d),D
FD7305	LD (IY+d),E
FD7405	LD (IY+d),H
FD7505	LD (IY+d),L
FD360520	LD (IY+d),N
328405	LD (NN),A
ED438405	LD (NN),BC
ED538405	LD (NN),DE
228405	LD (NN),HL
DD228405	LD (NN),IX
FD228405	LD (NN),IY
ED738405	LD (NN),SP
0A	LD A,(BC)
1A	LD A,(DE)
7E	LD A,(HL)
DD7E05	LD A,(IX+d)
FD7E05	LD A,(IY+d)
3A8405	LD A,(NN)
7F	LD A,A
78	LD A,B
79	LD A,C
7A	LD A,D
7B	LD A,E
7C	LD A,H
ED57	LD A,I
7D	LD A,L
3E20	LD A,N
46	LD B,(HL)
DD4605	LD B,(IX+d)
FD4605	LD B,(IY+d)
47	LD B,A
40	LD B,B
41	LD B,C
42	LD B,D
43	LD B,E
44	LD B,H,NN
45	LD B,L
0620	LD B,N
ED4B8405	LD BC,(NN)
018405	LD BC,NN
4E	LD C,(HL)

CBCC	SET 1,H
CBCD	SET 1,L
CBD6	SET 2,(HL)
DDCB05D6	SET 2,(IX+d)
FDCB05D6	SET 2,(IY+d)
CBD7	SET 2,A
CBD0	SET 2,B
CBD1	SET 2,C
CBD2	SET 2,D
CBD3	SET 2,E
CBD4	SET 2,H
CBD5	SET 2,L
CBD8	SET 3,B
CBDE	SET 3,(HL)
DDCB05DE	SET 3,(IX+d)
FDCB05DE	SET 3,(IY+d)
CBDF	SET 3,A
CBD9	SET 3,C
CBDA	SET 3,D
CBDB	SET 3,E
CBDC	SET 3,H
CBDD	SET 3,L
CBE6	SET 4,(HL)
DDCB05E6	SET 4,(IX+d)
FDCB05E6	SET 4,(IY+d)
CBE7	SET 4,A
CBE0	SET 4,B
CBE1	SET 4,C
CBE2	SET 4,D
CBE3	SET 4,E
CBE4	SET 4,H
CBE5	SET 4,L
CBEE	SET 5,(HL)
DDCB05EE	SET 5,(IX+d)
FDCB05EE	SET 5,(IY+d)
CBEF	SET 5,A
CBE8	SET 5,B
CBE9	SET 5,C
CBEA	SET 5,D
CBEB	SET 5,E
CBEC	SET 5,H
CBED	SET 5,L
CBF6	SET 6,(HL)
DDCB05F6	SET 6,(IX+d)
FDCB05F6	SET 6,(IY+d)
CBF7	SET 6,A
CBF0	SET 6,B
CBF1	SET 6,C

CBF2	SET 6,D
CBF3	SET 6,E
CBF4	SET 6,H
CBF5	SET 6,L
CBFE	SET 7,(HL)
DDCB05FE	SET 7,(IX+d)
FDCB05FE	SET 7,(IY+d)
CBFF	SET 7,A
CBF8	SET 7,B
CBF9	SET 7,C
CBFA	SET 7,D
CBFB	SET 7,E
CBFC	SET 7,H
CBFD	SET 7,L
CB26	SLA (HL)
DDCB0526	SLA (IX+d)
FDCB0526	SLA (IY+d)
CB27	SLA A
CB20	SLA B
CB21	SLA C
CB22	SLA D
CB23	SLA E
CB24	SLA H
CB25	SLA L
CB2E	SRA (HL)
DDCB052E	SRA (IX+d)
FDCB052E	SRA (IY+d)
CB2F	SRA A
CB28	SRA B
CB29	SRA C
CB2A	SRA D
CB2B	SRA E
CB2C	SRA H
CB2D	SRA L
CB3E	SRL (HL)
DDCB053E	SRL (IX+d)
FDCB053E	SRL (IY+d)
CB3F	SRL A
CB38	SRL B
CB39	SRL C
CB3A	SRL D
CB3B	SRL E
CB3C	SRL H
CB3D	SRL L
96	SUB (HL)
DD9605	SUB (IX+d)
FD9605	SUB (IY+d)
97	SUB A

90	SUB B
91	SUB C
92	SUB D
93	SUB E
94	SUB H
95	SUB L
D620	SUB N
AE	XOR (HL)
DDAE05	XOR (IX+d)
FDAE05	XOR (IY+d)
AF	XOR A
A8	XOR B
A9	XOR C
AA	XOR D
AB	XOR E
AC	XOR H
AD	XOR L
EE20	XOR N

Example Values

nn	EQU	584H
d	EQU	5
n	EQU	20H
e		30H

MAIN REG SET		ALTERNATE REG SET		
ACCUMULATOR A	FLAGS F	ACCUMULATOR A'	FLAGS F'	
B	C	B'	C'	GENERAL PURPOSE REGISTERS
D	E	D'	E'	
H	L	H'	L'	

INTERRUPT VECTOR I	MEMORY REFRESH R	SPECIAL PURPOSE REGISTERS
INDEX REGISTER IX		
INDEX REGISTER IY		
STACK POINTER SP		
PROGRAM COUNTER PC		

Z80-CPU
REGISTER CONFIGURATION

SUMMARY OF FLAG OPERATION

Instruction	D7 S	Z		H		P/V	N	D0 C	Comments
ADD s; ADC s	↕	↕	X	↕	X	V	0	↕	8-bit add or add with carry
SUB s; SBC s; CP s; NEG	↕	↕	X	↕	X	V	1	↕	8-bit subtract, subtract with carry, compare and negate accumulator
AND s	↕	↕	X	↕	X	P	0	0	Logical operations
OR s; XOR s	↕	↕	X	↕	X	P	0	0	Logical operations
INC s	↕	↕	X	↕	X	V	0	•	8-bit increment
DEC s	↕	↕	X	↕	X	V	1	•	8-bit decrement
ADD DD, SS	•	•	X	X	X	•	0	↕	16-bit add
ADC HL, SS	↕	↕	X	X	X	V	0	↕	16-bit add with carry
SBC HL, SS	↕	↕	X	X	X	V	1	↕	16-bit subtract with carry
RLA; RLCA; RRA; RRCA	•	•	X	0	X	•	0	↕	Rotate accumulator
RL s; RLC s; RR s; RRC s; SLA s; SRA s; SRL s	↕	↕	X	0	X	P	0	↕	Rotate and shift locations
RLD; RRD	↕	↕	X	0	X	P	0	•	Rotate digit left and right
DAA	↕	↕	X	↕	X	P	•	↕	Decimal adjust accumulator
CPL	•	•	X	1	X	•	1	•	Complement accumulator
SCF	•	•	X	0	X	•	0	1	Set carry
CCF	•	•	X	X	X	•	0	↕	Complement carry
IN r, (C)	↕	↕	X	0	X	P	0	•	Input register indirect
INI; IND; OUTI; OUTD	X	↕	X	X	X	X	1	•	Block input and output Z = 0 if B ≠ 0 otherwise Z = 1
INIR; INDR; OTIR; OTDR	X	1	X	X	X	X	1	•	Block input and output Z = 0 if B ≠ 0 otherwise Z = 1
LDI; LDD	X	X	X	0	X	↕	0	•	Block transfer instructions P/V = 1 if BC ≠ 0, otherwise P/V = 0
LDIR; LDDR	X	X	X	0	X	0	0	•	Block transfer instructions P/V = 1 if BC ≠ 0, otherwise P/V = 0
CPI; CPIR; CPD; CPDR	X	↕	X	X	X	↕	1	•	Block search instructions Z = 1 if A = (HL), otherwise Z = 0 P/V = 1 if BC ≠ 0, otherwise P/V = 0
LD A, I; LD A, R	↕	↕	X	0	X	IFF	0	•	The content of the interrupt enable flip-flop (IFF) is copied into the P/V flag
BIT b, s	X	↕	X	1	X	X	0	•	The state of bit b of location s is copied into the Z flag

The following notation is used in this table:

Symbol	Operation
C	Carry/link flag. C=1 if the operation produced a carry from the MSB of the operand or result.
Z	Zero flag. Z=1 if the result of the operation is zero.
S	Sign flag. S=1 if the MSB of the result is one.
P/V	Parity or overflow flag. Parity (P) and overflow (V) share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with the overflow of the result. If P/V holds parity, P/V=1 if the result of the operation is even, P/V=0 if result is odd. If P/V holds overflow, P/V=1 if the result of the operation produced an overflow
H	Half-carry flag. H=1 if the add or subtract operation produced a carry into or borrow from bit 4 of the accumulator
N	Add/Subtract flag. N=1 if the previous operation was a subtract. H and N flags are used in conjunction with the decimal adjust instruction (DAA) to properly correct the result into packed BCD format following addition or subtraction using operands with packed BCD format.
↕	The flag is affected according to the result of the operation.
•	The flag is unchanged by the operation.
0	The flag is reset by the operation.
1	The flag is set by the operation.
X	The flag is a "don't care".
V	P/V flag affected according to the overflow result of the operation.
P	P/V flag affected according to the parity result of the operation.
r	Any one of the CPU registers A, B, C, D, E, H, L.
s	Any 8-bit location for all the addressing modes allowed for the particular instruction.
ss	Any 16-bit location for all the addressing modes allowed for that instruction.
ii	Any one of the two index registers IX or IY.
R	Refresh counter.
n	8-bit value in range <0, 255>
nn	16-bit value in range <0, 65535>

8-BIT LOAD GROUP
'LD'

DESTINATION		SOURCE: IMPLIED I	IMPLIED R	REGISTER A	REGISTER B	REGISTER C	REGISTER D	REGISTER E	REGISTER H	REGISTER L	REG INDIRECT (HL)	REG INDIRECT (BC)	REG INDIRECT (DE)	INDEXED (IX+d)	INDEXED (IY+d)	EXT ADDR (nn)	IMMEDIATE n
REGISTER	A	ED 57	ED 5F	7F	78	79	7A	7B	7C	7D	7E	0A	1A	DD 7E d	FD 7E d	3A n n	3E n
	B			47	40	41	42	43	44	45	46			DD 46 d	FD 46 d		06 n
	C			4F	48	49	4A	4B	4C	4D	4E			DD 4E d	FD 4E d		0E n
	D			57	50	51	52	53	54	55	56			DD 56 d	FD 56 d		16 n
	E			5F	58	59	5A	5B	5C	5D	5E			DD 5E d	FD 5E d		1E n
	H			67	60	61	62	63	64	65	66			DD 66 d	FD 66 d		26 n
	L			6F	68	69	6A	6B	6C	6D	6E			DD 6E d	FD 6E d		2E n
REG INDIRECT	(HL)			77	70	71	72	73	74	75							36 n
	(BC)			02													
	(DE)			12													
INDEXED	(IX+d)			DD 77 d	DD 70 d	DD 71 d	DD 72 d	DD 73 d	DD 74 d	DD 75 d							DD 36 d n
	(IY+d)			FD 77 d	FD 70 d	FD 71 d	FD 72 d	FD 73 d	FD 74 d	FD 75 d							FD 36 d n
EXT.ADDR.	(nn)			32 n n													
IMPLIED	I			ED 47													
	R			ED 4F													

8-BIT LOAD GROUP

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments r, s	Reg.
LD r, s	r — s	•	•	X	•	X	•	•	•	01 r s		1	1	4	r, s	Reg.
LD r, n	r — n	•	•	X	•	X	•	•	•	00 r 110		2	2	7	000	B
										— n —					001	C
LD r, (HL)	r — (HL)	•	•	X	•	X	•	•	•	01 r 110		1	2	7	010	D
LD r, (IX+d)	r — (IX+d)	•	•	X	•	X	•	•	•	11 011 101	DD	3	5	19	011	E
										01 r 110					100	H
										— d —					101	L
LD r, (IY+d)	r — (IY+d)	•	•	X	•	X	•	•	•	11 111 101	FD	3	5	19	111	A
										01 r 110						
										— d —						
LD (HL), r	(HL) — r	•	•	X	•	X	•	•	•	01 110 r		1	2	7		
LD (IX+d), r	(IX+d) — r	•	•	X	•	X	•	•	•	11 011 101	DD	3	5	19		
										01 110 r						
										— d —						
LD (IY+d), r	(IY+d) — r	•	•	X	•	X	•	•	•	11 111 101	FD	3	5	19		
										01 110 r						
										— d —						
LD (HL), n	(HL) — n	•	•	X	•	X	•	•	•	00 110 110	36	2	3	10		
										— n —						
LD (IX+d), n	(IX+d) — n	•	•	X	•	X	•	•	•	11 011 101	DD	4	5	19		
										00 110 110	36					
										— d —						
										— n —						
LD (IY+d), n	(IY+d) — n	•	•	X	•	X	•	•	•	11 111 101	FD	4	5	19		
										00 110 110	36					
										— d —						
										— n —						
LD A, (BC)	A — (BC)	•	•	X	•	X	•	•	•	00 001 010	0A	1	2	7		
LD A, (DE)	A — (DE)	•	•	X	•	X	•	•	•	00 011 010	1A	1	2	7		
LD A, (nn)	A — (nn)	•	•	X	•	X	•	•	•	00 111 010	3A	3	4	13		
										— n —						
										— n —						
LD (BC), A	(BC) — A	•	•	X	•	X	•	•	•	00 000 010	02	1	2	7		
LD (DE), A	(DE) — A	•	•	X	•	X	•	•	•	00 010 010	12	1	2	7		
LD (nn), A	(nn) — A	•	•	X	•	X	•	•	•	00 110 010	32	3	4	13		
										— n —						
										— n —						
LD A, I	A — I	↕	↕	X	0	X	IFF	0	•	11 101 101	ED	2	2	9		
										01 010 111	57					
LD A, R	A — R	↕	↕	X	0	X	IFF	0	•	11 101 101	ED	2	2	9		
										01 011 111	5F					
LD I, A	I — A	•	•	X	•	X	•	•	•	11 101 101	ED	2	2	9		
										01 000 111	47					
LD R, A	R — A	•	•	X	•	X	•	•	•	11 101 101	ED	2	2	9		
										01 001 111	4F					

Notes: r, s means any of the registers A, B, C, D, E, H, L
IFF the content of the interrupt enable flip-flop (IFF) is copied into the P/V flag

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown.
↕ = flag is affected according to the result of the operation.

16-BIT LOAD GROUP
'LD'
PUSH' AND 'POP'

DESTINATION		SOURCE: REGISTER AF	BC	DE	HL	SP	IX	IY	IMM. EXT. nn	EXT. ADDR. (nn)	REG. INDIR. (SP)
REGISTER	AF										F1
REGISTER	BC								01 n n	ED 4B n n	C1
REGISTER	DE								11 n n	ED 5B n n	D1
REGISTER	HL								21 n n	2A n n	E1
REGISTER	SP				F9		DD F9	FD F9	31 n n	ED 7B n n	
REGISTER	IX								DD 21 n n	DD 2A n n	DD E1
REGISTER	IY								FD 21 n n	FD 2A n n	FD E1
EXT. ADDR.	(nn)		ED 43 n n	ED 53 n n	22 n n	ED 73 n n	DD 22 n n	FD 22 n n			
REG. IND.	(SP)	F5	C5	D5	E5		DD E5	FD E5			

PUSH INSTRUCTIONS

POP INSTRUCTIONS

NOTE: The Push & Pop Instructions adjust the SP after every execution.

16-BIT LOAD GROUP

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
LD dd, nn	dd ← nn	•	•	X	•	X	•	•	•	00 dd0 001 – n – – n –		3	3	10	dd Pair 00 BC 01 DE 10 HL 11 SP
LD IX, nn	IX ← nn	•	•	X	•	X	•	•	•	11 011 101 00 100 001 – n – – n –	DD 21	4	4	14	
LD IY, nn	IY ← nn	•	•	X	•	X	•	•	•	11 111 101 00 100 001 – n – – n –	FD 21	4	4	14	
LD HL, (nn)	H ← (nn+1) L ← (nn)	•	•	X	•	X	•	•	•	00 101 010 – n – – n –	2A	3	5	16	
LD dd, (nn)	dd_H ← (nn+1) dd_L ← (nn)	•	•	X	•	X	•	•	•	11 101 101 01 dd1 011 – n – – n –	ED	4	6	20	
LD IX, (nn)	IX_H ← (nn+1) IX_L ← (nn)	•	•	X	•	X	•	•	•	11 011 101 00 101 010 – n – – n –	DD 2A	4	6	20	
LD IY, (nn)	IY_H ← (nn+1) IY_L ← (nn)	•	•	X	•	X	•	•	•	11 111 101 00 101 010 – n – – n –	FD 2A	4	6	20	
LD (nn), HL	(nn+1) ← H (nn) ← L	•	•	X	•	X	•	•	•	00 100 010 – n – – n –	22	3	5	16	
LD (nn), dd	(nn+1) ← dd_H (nn) ← dd_L	•	•	X	•	X	•	•	•	11 101 101 01 dd0 011 – n – – n –	ED	4	6	20	
LD (nn), IX	(nn+1) ← IX_H (nn) ← IX_L	•	•	X	•	X	•	•	•	11 011 101 00 100 010 – n – – n –	DD 22	4	6	20	
LD (nn), IY	(nn+1) ← IY_H (nn) ← IY_L	•	•	X	•	X	•	•	•	11 111 101 00 100 010 – n – – n –	FD 22	4	6	20	
LD SP, HL	SP ← HL	•	•	X	•	X	•	•	•	11 111 001	F9	1	1	6	
LD SP, IX	SP ← IX	•	•	X	•	X	•	•	•	11 011 101 11 111 001	DD F9	2	2	10	
LD SP, IY	SP ← IY	•	•	X	•	X	•	•	•	11 111 101 11 111 001	FD F9	2	2	10	
PUSH qq	(SP-2) ← qq_L (SP-1) ← qq_H	•	•	X	•	X	•	•	•	11 qq0 101		1	3	11	qq Pair 00 BC 01 DE 10 HL 11 AF
PUSH IX	(SP-2) ← IX_L (SP-1) ← IX_H	•	•	X	•	X	•	•	•	11 011 101 11 100 101	DD E5	2	4	15	
PUSH IY	(SP-2) ← IY_L (SP-1) ← IY_H	•	•	X	•	X	•	•	•	11 111 101 11 100 101	FD E5	2	4	15	
POP qq	qq_H ← (SP+1) qq_L ← (SP)	•	•	X	•	X	•	•	•	11 qq0 001		1	3	10	
POP IX	IX_H ← (SP+1) IX_L ← (SP)	•	•	X	•	X	•	•	•	11 011 101 11 100 001	DD E1	2	4	14	
POP IY	IY_H ← (SP+1) IY_L ← (SP)	•	•	X	•	X	•	•	•	11 111 101 11 100 001	FD E1	2	4	14	

Notes: dd is any of the register pairs BC, DE, HL, SP
qq is any of the register pairs AF, BC, DE, HL
$(PAIR)_H$, $(PAIR)_L$ refer to high order and low order eight bits of the register pair respectively.
e.g. BC_L = C, AF_H = A

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknow
↕ flag is affected according to the result of the operation

EXCHANGES
'EX' AND 'EXX'

		IMPLIED ADDRESSING				
		AF'	BC', DE' & HL'	HL	IX	IY
IMPLIED	AF	08				
	BC, DE & HL		D9			
	DE			EB		
REG. INDIR.	(SP)			E3	DD E3	FD E3

BLOCK TRANSFER GROUP

		SOURCE	
		REG. INDIR.	
		(HL)	
DESTINATION REG. INDIR.	(DE)	ED A0	'LDI' – Load (DE) ← (HL) Inc HL & DE, Dec BC
		ED B0	'LDIR' – Load (DE)←(HL) Inc HL & DE, Dec BC, Repeat until BC = 0
		ED A8	'LDD' – Load (DE) ← (HL) Dec HL & DE, Dec BC
		ED B8	'LDDR' – Load (DE)←(HL) Dec HL & DE, Dec BC, Repeat until BC = 0

HL points to source
DE points to destination
BC is byte counter

BLOCK SEARCH GROUP

SEARCH LOCATION	
REG. INDIR.	
(HL)	
ED A1	'CPI' Inc HL, Dec BC
ED B1	'CPIR' – Inc HL, Dec BC repeat until BC = 0 or find match
ED A9	'CPD' – Dec HL & BC
ED B9	'CPDR' – Dec HL & BC Repeat until BC = 0 or find match

HL points to location in memory to be compared with accumulator contents
BC is byte counter

EXCHANGE GROUP AND BLOCK TRANSFER AND SEARCH GROUP

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
EX DE, HL	DE—HL	•	•	X	•	X	•	•	•	11 101 011	EB	1	1	4	
EX AF, AF'	AF—AF'	•	•	X	•	X	•	•	•	00 001 000	08	1	1	4	
EXX	BC—BC' DE—DE' HL—HL'	•	•	X	•	X	•	•	•	11 011 001	D9	1	1	4	Register bank and auxiliary register bank exchange
EX (SP), HL	H —(SP+1) L —(SP)	•	•	X	•	X	•	•	•	11 100 011	E3	1	5	19	
EX (SP), IX	IX_H—(SP+1) IX_L—(SP)	•	•	X	•	X	•	•	•	11 011 101 11 100 011	DD E3	2	6	23	
EX (SP), IY	IY_H—(SP+1) IY_L—(SP)	•	•	X	•	X	•	•	•	11 111 101 11 100 011	FD E3	2	6	23	
LDI	(DE)—(HL) DE — DE+1 HL — HL+1 BC — BC-1	•	•	X	0	X	① ↕	0	•	11 101 101 10 100 000	ED A0	2	4	16	Load (HL) into (DE), increment the pointers and decrement the byte counter (BC)
LDIR	(DE)—(HL) DE — DE+1 HL — HL+1 BC — BC-1 Repeat until BC = 0	•	•	X	0	X	0	0	•	11 101 101 10 110 000	ED B0	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
LDD	(DE)—(HL) DE — DE-1 HL — HL-1 BC — BC-1	•	•	X	0	X	① ↕	0	•	11 101 101 10 101 000	ED A8	2	4	16	
LDDR	(DE)—(HL) DE — DE-1 HL — HL-1 BC — BC-1 Repeat until BC = 0	•	•	X	0	X	0	0	•	11 101 101 10 111 000	ED B8	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
CPI	A – (HL) HL — HL+1 BC — BC-1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 100 001	ED A1	2	4	16	
CPIR	A – (HL) HL — HL+1 BC — BC-1 Repeat until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 110 001	ED B1	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)
CPD	A – (HL) HL — HL-1 BC — BC-1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 101 001	ED A9	2	4	16	
CPDR	A – (HL) HL — HL-1 BC — BC-1 Repeat until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 111 001	ED B9	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)

Notes: ① P/V flag is 0 if the result of BC-1 = 0, otherwise P/V = 1
② Z flag is 1 if A = (HL), otherwise Z = 0.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

8-BIT ARITHMETIC AND LOGIC

SOURCE

	REGISTER ADDRESSING							REG. INDIR.	INDEXED		IMMED.
	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)	n
'ADD'	87	80	81	82	83	84	85	86	DD 86 d	FD 86 d	C6 n
ADD w CARRY 'ADC'	8F	88	89	8A	8B	8C	8D	8E	DD 8E d	FD 8E d	CE n
SUBTRACT 'SUB'	97	90	91	92	93	94	95	96	DD 96 d	FD 96 d	D6 n
SUB w CARRY 'SBC'	9F	98	99	9A	9B	9C	9D	9E	DD 9E d	FD 9E d	DE n
'AND'	A7	A0	A1	A2	A3	A4	A5	A6	DD A6 d	FD A6 d	E6 n
'XOR'	AF	A8	A9	AA	AB	AC	AD	AE	DD AE d	FD AE d	EE n
'OR'	B7	B0	B1	B2	B3	B4	B5	B6	DD B6 d	FD B6 d	F6 n
COMPARE 'CP'	BF	B8	B9	BA	BB	BC	BD	BE	DD BE d	FD BE d	FE n
INCREMENT 'INC'	3C	04	0C	14	1C	24	2C	34	DD 34 d	FD 34 d	
DECREMENT 'DEC'	3D	05	0D	15	1D	25	2D	35	DD 35 d	FD 35 d	

8-BIT ARITHMETIC AND LOGICAL GROUP

Mnemonic	Symbolic Operation	Flags S	Z	P	H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
ADD A, r	A ← A + r	↕	↕	X	↕	X	V	0	↕	10 000 r		1	1	4	r Reg.
ADD A, n	A ← A + n	↕	↕	X	↕	X	V	0	↕	11 000 110 ← n →		2	2	7	000 B 001 C 010 D
ADD A, (HL)	A ← A+(HL)	↕	↕	X	↕	X	V	0	↕	10 000 110		1	2	7	011 E
ADD A, (IX+d)	A←A+(IX+d)	↕	↕	X	↕	X	V	0	↕	11 011 101 10 000 110 ← d →	DD	3	5	19	100 H 101 L 111 A
ADD A, (IY+d)	A←A+(IY+d)	↕	↕	X	↕	X	V	0	↕	11 111 101 10 000 110 ← d →	FD	3	5	19	
ADC A, s	A←A+s+CY	↕	↕	X	↕	X	V	0	↕	001					s is any of r, n,
SUB s	A←A−s	↕	↕	X	↕	X	V	1	↕	010					(HL), (IX+d),
SBC A, s	A←A−s−CY	↕	↕	X	↕	X	V	1	↕	011					(IY+d) as shown for
AND s	A←A ∧ s	↕	↕	X	1	X	P	0	0	100					ADD instruction.
OR s	A←A ∨ s	↕	↕	X	↕	X	P	0	0	110					The indicated bits
XOR s	A←A ⊕ s	↕	↕	X	↕	X	P	0	0	101					replace the 000 in
CP s	A−s	↕	↕	X	↕	X	V	1	↕	111					the ADD set above.
INC r	r ← r + 1	↕	↕	X	↕	X	V	0	•	00 r 100		1	1	4	
INC (HL)	(HL)←(HL)+1	↕	↕	X	↕	X	V	0	•	00 110 100		1	3	11	
INC (IX+d)	(IX+d) ← (IX+d)+1	↕	↕	X	↕	X	V	0	•	11 011 101 00 110 100 ← d →	DD	3	6	23	
INC (IY+d)	(IY+d) ← (IY+d)+1	↕	↕	X	↕	X	V	0	•	11 111 101 00 110 100 ← d →	FD	3	6	23	
DEC s	s ← s − 1	↕	↕	X	↕	X	V	1	•	101					s is any of r, (HL), (IX+d), (IY+d) as shown for INC. DEC same format and states as INC. Replace 100 with 101 in OP Code.

Notes: The V symbol in the P/V flag column indicates that the P/V flag contains the overflow of the result of the operation. Similarly the P symbol indicates parity. V = 1 means overflow, V = 0 means not overflow, P = 1 means parity of the result is even, P = 0 means parity of the result is odd.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown.
↕ = flag is affected according to the result of the operation.

GENERAL PURPOSE AF OPERATIONS

Decimal Adjust Acc, 'DAA'	27
Complement Acc, 'CPL'	2F
Negate Acc, 'NEG' (2's complement)	ED 44
Complement Carry Flag, 'CCF'	3F
Set Carry Flag, 'SCF'	37

MISCELLANEOUS CPU CONTROL

'NOP'	00	
'HALT'	76	
DISABLE INT '(DI)'	F3	
ENABLE INT '(EI)'	FB	
SET INT MODE 0 'IM 0'	ED 46	8080A MODE
SET INT MODE 1 'IM 1'	ED 56	RESTART TO LOCATION 0038_H
SET INT MODE 2 'IM 2'	ED 5E	INDIRECT CALL USING REGISTER I AND 8 BITS FROM INTERRUPTING DEVICE AS A POINTER.

GENERAL PURPOSE ARITHMETIC AND CPU CONTROL GROUPS

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
DAA	Converts acc. content into packed BCD following add or subtract with packed BCD operands	↕	↕	X	↕	X	P	•	↕	00 100 111	27	1	1	4	Decimal adjust accumulator
CPL	$A - \overline{A}$	•	•	X	1	X	•	1	•	00 101 111	2F	1	1	4	Complement accumulator (One's complement)
NEG	$A - \overline{A}+1$	↕	↕	X	↕	X	V	1	↕	11 101 101 01 000 100	ED 44	2	2	8	Negate acc. (two's complement)
CCF	$CY - \overline{CY}$	•	•	X	X	X	•	0	↕	00 111 111	3F	1	1	4	Complement carry flag
SCF	CY - 1	•	•	X	0	X	•	0	1	00 110 111	37	1	1	4	Set carry flag
NOP	No operation	•	•	X	•	X	•	•	•	00 000 000	00	1	1	4	
HALT	CPU halted	•	•	X	•	X	•	•	•	01 110 110	76	1	1	4	
DI	IFF - 0	•	•	X	•	X	•	•	•	11 110 011	F3	1	1	4	
EI	IFF - 1	•	•	X	•	X	•	•	•	11 111 011	FB	1	1	4	
IM 0	Set interrupt mode 0	•	•	X	•	X	•	•	•	11 101 101 01 000 110	ED 46	2	2	8	
IM 1	Set interrupt mode 1	•	•	X	•	X	•	•	•	11 101 101 01 010 110	ED 56	2	2	8	
IM 2	Set interrupt mode 2	•	•	X	•	X	•	•	•	11 101 101 01 011 110	ED 5E	2	2	8	

Notes: IFF indicates the interrupt enable flip-flop
CY indicates the carry flip-flop.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

16-BIT ARITHMETIC

DESTINATION		SOURCE: BC	DE	HL	SP	IX	IY
'ADD'	HL	09	19	29	39		
	IX	DD 09	DD 19		DD 39	DD 29	
	IY	FD 09	FD 19		FD 39		FD 29
ADD WITH CARRY AND SET FLAGS 'ADC'	HL	ED 4A	ED 5A	ED 6A	ED 7A		
SUB WITH CARRY AND SET FLAGS 'SBC'	HL	ED 42	ED 52	ED 62	ED 72		
INCREMENT 'INC'		03	13	23	33	DD 23	FD 23
DECREMENT 'DEC'		0B	1B	2B	3B	DD 2B	FD 2B

16-BIT ARITHMETIC GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
ADD HL, ss	HL ← HL+ss	•	•	X	X	X	•	0	↕	00 ss1 001		1	3	11	ss Reg. 00 BC
ADC HL, ss	HL ← HL+ss+CY	↕	↕	X	X	X	V	0	↕	11 101 101 01 ss1 010	ED	2	4	15	01 DE 10 HL 11 SP
SBC HL, ss	HL ← HL-ss-CY	↕	↕	X	X	X	V	1	↕	11 101 101 01 ss0 010	ED	2	4	15	
ADD IX, pp	IX ← IX + pp	•	•	X	X	X	•	0	↕	11 011 101 00 pp1 001	DD	2	4	15	pp Reg. 00 BC 01 DE 10 IX 11 SP
ADD IY, rr	IY ← IY + rr	•	•	X	X	X	•	0	↕	11 111 101 00 rr1 001	FD	2	4	15	rr Reg. 00 BC 01 DE 10 IY 11 SP
INC ss	ss ← ss + 1	•	•	X	•	X	•	•	•	00 ss0 011		1	1	6	
INC IX	IX ← IX + 1	•	•	X	•	X	•	•	•	11 011 101 00 100 011	DD 23	2	2	10	
INC IY	IY ← IY + 1	•	•	X	•	X	•	•	•	11 111 101 00 100 011	FD 23	2	2	10	
DEC ss	ss ← ss - 1	•	•	X	•	X	•	•	•	00 ss1 011		1	1	6	
DEC IX	IX ← IX - 1	•	•	X	•	X	•	•	•	11 011 101 00 101 011	DD 2B	2	2	10	
DEC IY	IY ← IY - 1	•	•	X	•	X	•	•	•	11 111 101 00 101 011	FD 2B	2	2	10	

Notes: ss is any of the register pairs BC, DE, HL, SP
pp is any of the register pairs BC, DE, IX, SP
rr is any of the register pairs BC, DE, IY, SP.

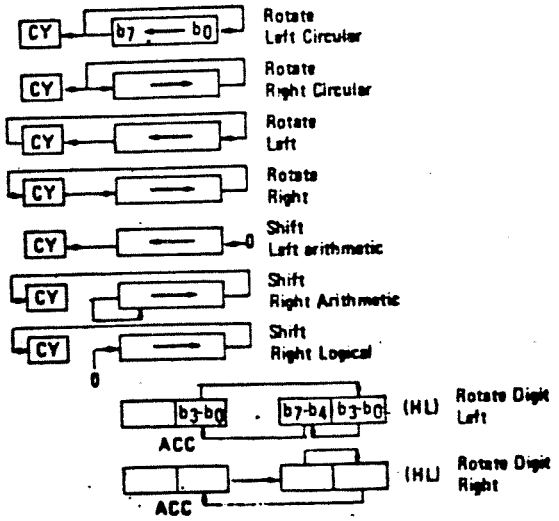
Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown.
↕ = flag is affected according to the result of the operation.

ROTATES AND SHIFTS

Source and Destination

TYPE OF ROTATE OR SHIFT	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)
'RLC'	CB 07	CB 00	CB 01	CB 02	CB 03	CB 04	CB 05	CB 06	DD CB d 06	FD CB d 06
'RRC'	CB 0F	CB 08	CB 09	CB 0A	CB 0B	CB 0C	CB 0D	CB 0E	DD CB d 0E	FD CB d 0E
'RL'	CB 17	CB 10	CB 11	CB 12	CB 13	CB 14	CB 15	CB 16	DD CB d 16	FD CB d 16
'RR'	CB 1F	CB 18	CB 19	CB 1A	CB 1B	CB 1C	CB 1D	CB 1E	DD CB d 1E	FD CB d 1E
'SLA'	CB 27	CB 20	CB 21	CB 22	CB 23	CB 24	CB 25	CB 26	DD CB d 26	FD CB d 26
'SRA'	CB 2F	CB 28	CB 29	CB 2A	CB 2B	CB 2C	CB 2D	CB 2E	DD CB d 2E	FD CB d 2E
'SRL'	CB 3F	CB 38	CB 39	CB 3A	CB 3B	CB 3C	CB 3D	CB 3E	DD CB d 3E	FD CB d 3E
'RLD'								ED 6F		
'RRD'								ED 67		

	A
'RLCA'	07
'RRCA'	0F
'RLA'	17
'RRA'	1F



ROTATE AND SHIFT GROUP

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
RLCA	CY ← 7 ← 0 ← A	•	•	X	0	X	•	0	↕	00 000 111	07	1	1	4	Rotate left circular accumulator
RLA	CY ← 7 ← 0 ← A	•	•	X	0	X	•	0	↕	00 010 111	17	1	1	4	Rotate left accumulator
RRCA	7 → 0 → CY, A	•	•	X	0	X	•	0	↕	00 001 111	0F	1	1	4	Rotate right circular accumulator
RRA	7 → 0 → CY, A	•	•	X	0	X	•	0	↕	00 011 111	1F	1	1	4	Rotate right accumulator
RLC r		↕	↕	X	0	X	P	0	↕	11 001 011 00 000 r	CB	2	2	8	Rotate left circular register r
RLC (HL)		↕	↕	X	0	X	P	0	↕	11 001 011 00 000 110	CB	2	4	15	r Reg. 000 B 001 C 010 D 011 E 100 H 101 L 111 A
RLC (IX+d)	CY ← 7 ← 0 r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	11 011 101 11 001 011 ← d → 00 000 110	DD CB	4	6	23	
RLC (IY+d)		↕	↕	X	0	X	P	0	↕	11 111 101 11 001 011 ← d → 00 000 110	FD CB	4	6	23	
RL s	CY ← 7 ← 0 s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	010					Instruction format and states are as shown for RLC's. To form new Op-Code replace 000 of RLC's with shown code
RRC s	7 → 0 → CY s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	001					
RR s	7 → 0 → CY s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	011					
SLA s	CY ← 7 ← 0 ← 0 s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	100					
SRA s	7 → 0 → CY s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	101					
SRL s	0 → 7 → 0 → CY s ≡ r,(HL),(IX+d),(IY+d)	↕	↕	X	0	X	P	0	↕	111					
RLD	A 7-4 3-0 7-4 3-0 (HL)	↕	↕	X	0	X	P	0	•	11 101 101 01 101 111	ED 6F	2	5	18	Rotate digit left and right between the accumulator and location (HL).
RRD	A 7-4 3-0 7-4 3-0 (HL)	↕	↕	X	0	X	P	0	•	11 101 101 01 100 111	ED 67	2	5	18	The content of the upper half of the accumulator is unaffected

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown.
↕ = flag is affected according to the result of the operation.

BIT MANIPULATION GROUP

		REGISTER ADDRESSING							REG. INDIR.	INDEXED	
	BIT	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)
TEST 'BIT'	0	CB 47	CB 40	CB 41	CB 42	CB 43	CB 44	CB 45	CB 46	DD CB d 46	FD CB d 46
	1	CB 4F	CB 48	CB 49	CB 4A	CB 4B	CB 4C	CB 4D	CB 4E	DD CB d 4E	FD CB d 4E
	2	CB 57	CB 50	CB 51	CB 52	CB 53	CB 54	CB 55	CB 56	DD CB d 56	FD CB d 56
	3	CB 5F	CB 58	CB 59	CB 5A	CB 5B	CB 5C	CB 5D	CB 5E	DD CB d 5E	FD CB d 5E
	4	CB 67	CB 60	CB 61	CB 62	CB 63	CB 64	CB 65	CB 66	DD CB d 66	FD CB d 66
	5	CB 6F	CB 68	CB 69	CB 6A	CB 6B	CB 6C	CB 6D	CB 6E	DD CB d 6E	FD CB d 6E
	6	CB 77	CB 70	CB 71	CB 72	CB 73	CB 74	CB 75	CB 76	DD CB d 76	FD CB d 76
	7	CB 7F	CB 78	CB 79	CB 7A	CB 7B	CB 7C	CB 7D	CB 7E	DD CB d 7E	FD CB d 7E
RESET BIT 'RES'	0	CB 87	CB 80	CB 81	CB 82	CB 83	CB 84	CB 85	CB 86	DD CB d 86	FD CB d 86
	1	CB 8F	CB 88	CB 89	CB 8A	CB 8B	CB 8C	CB 8D	CB 8E	DD CB d 8E	FD CB d 8E
	2	CB 97	CB 90	CB 91	CB 92	CB 93	CB 94	CB 95	CB 96	DD CB d 96	FD CB d 96
	3	CB 9F	CB 98	CB 99	CB 9A	CB 9B	CB 9C	CB 9D	CB 9E	DD CB d 9E	FD CB d 9E
	4	CB A7	CB A0	CB A1	CB A2	CB A3	CB A4	CB A5	CB A6	DD CB d A6	FD CB d A6
	5	CB AF	CB A8	CB A9	CB AA	CB AB	CB AC	CB AD	CB AE	DD CB d AE	FD CB d AE
	6	CB B7	CB B0	CB B1	CB B2	CB B3	CB B4	CB B5	CB B6	DD CB d B6	FD CB d B6
	7	CB BF	CB B8	CB B9	CB BA	CB BB	CB BC	CB BD	CB BE	DD CB d BE	FD CB d BE
SET BIT 'SET'	0	CB C7	CB C0	CB C1	CB C2	CB C3	CB C4	CB C5	CB C6	DD CB d C6	FD CB d C6
	1	CB CF	CB C8	CB C9	CB CA	CB CB	CB CC	CB CD	CB CE	DD CB d CE	FD CB d CE
	2	CB D7	CB D0	CB D1	CB D2	CB D3	CB D4	CB D5	CB D6	DD CB d D6	FD CB d D6
	3	CB DF	CB D8	CB D9	CB DA	CB DB	CB DC	CB DD	CB DE	DD CB d DE	FD CB d DE
	4	CB E7	CB E0	CB E1	CB E2	CB E3	CB E4	CB E5	CB E6	DD CB d E6	FD CB d E6
	5	CB EF	CB E8	CB E9	CB EA	CB EB	CB EC	CB ED	CB EE	DD CB d EE	FD CB d EE
	6	CB F7	CB F0	CB F1	CB F2	CB F3	CB F4	CB F5	CB F6	DD CB d F6	FD CB d F6
	7	CB FF	CB F8	CB F9	CB FA	CB FB	CB FC	CB FD	CB FE	DD CB d FE	FD CB d FE

BIT SET, RESET AND TEST GROUP

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Op-Code 76 543 218	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
BIT b, r	$Z = \overline{r_b}$	X	‡	X	1	X	X	0	•	11 001 011 01 b r	CB	2	2	8	
BIT b, (HL)	$Z = \overline{(HL)_b}$	X	‡	X	1	X	X	0	•	11 001 011 01 b 110	CB	2	3	12	
BIT b, (IX+d)	$Z = \overline{(IX+d)_b}$	X	‡	X	1	X	X	0	•	11 011 101 11 001 011 – d – 01 b 110	DD CB	4	5	20	
BIT b, (IY+d)	$Z = \overline{(IY+d)_b}$	X	‡	X	1	X	X	0	•	11 111 101 11 001 011 – d – 01 b 110	FD CB	4	5	20	
SET b, r	$r_b = 1$	•	•	X	•	X	•	•	•	11 001 011 11 b r	CB	2	2	8	
SET b, (HL)	$(HL)_b = 1$	•	•	X	•	X	•	•	•	11 001 011 11 b 110	CB	2	4	15	
SET b, (IX+d)	$(IX+d)_b = 1$	•	•	X	•	X	•	•	•	11 011 101 11 001 011 – d – 11 b 110	DD CB	4	6	23	
SET b, (IY+d)	$(IY+d)_b = 1$	•	•	X	•	X	•	•	•	11 111 101 11 001 011 – d – 11 b 110	FD CB	4	6	23	
RES b, s	$s_b = 0$ $s \equiv r, (HL), (IX+d), (IY+d)$									10					To form new Op-Code replace 11 of SET b, s with 10. Flags and time states for SET instruction

r	Reg.
000	B
001	C
010	D
011	E
100	H
101	L
111	A

b	Bit Tested
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Notes: The notation s_b indicates bit b (0 to 7) of location s.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
‡ = flag is affected according to the result of the operation.

JUMP GROUP

CONDITION

			UN-COND.	CARRY	NON CARRY	ZERO	NON ZERO	PARITY EVEN	PARITY ODD	SIGN NEG.	SIGN POS.	REG. B≠0
JUMP 'JP'	IMMED. EXT.	nn	C3 n n	DA n n	D2 n n	CA n n	C2 n n	EA n n	E2 n n	FA n n	F2 n n	
JUMP 'JR'	RELATIVE	PC+e	18 e-2	38 e-2	30 e-2	28 e-2	20 e-2					
JUMP 'JP'	REG. INDIR.	(HL)	E9									
JUMP 'JP'		(IX)	DD E9									
JUMP 'JP'		(IY)	FD E9									
DECREMENT B, JUMP IF NON ZERO 'DNJZ'	RELATIVE	PC+e										10 e-2

JUMP GROUP

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
JP nn	PC ← nn	•	•	X	•	X	•	•	•	11 000 011 ← n ← ← n ←	C3	3	3	10	
JP cc, nn	If condition cc is true PC ← nn, otherwise continue	•	•	X	•	X	•	•	•	11 cc 010 ← n ← ← n ←		3	3	10	cc: 000 NZ non zero; 001 Z zero; 010 NC non carry; 011 C carry; 100 PO parity odd; 101 PE parity even; 110 P sign positive; 111 M sign negative
JR e	PC ← PC + e	•	•	X	•	X	•	•	•	00 011 000 ← e-2 ←	18	2	3	12	
JR C, e	If C = 0, continue	•	•	X	•	X	•	•	•	00 111 000 ← e-2 ←	38	2	2	7	If condition not met
	If C = 1, PC ← PC+e											2	3	12	If condition is met
JR NC, e	If C = 1, continue	•	•	X	•	X	•	•	•	00 110 000 ← e-2 ←	30	2	2	7	If condition not met
	If C = 0, PC ← PC+e											2	3	12	If condition is met
JR Z, e	If Z = 0 continue	•	•	X	•	X	•	•	•	00 101 000 ← e-2 ←	28	2	2	7	If condition not met
	If Z = 1, PC ← PC+e											2	3	12	If condition is met
JR NZ, e	If Z = 1, continue	•	•	X	•	X	•	•	•	00 100 000 ← e-2 ←	20	2	2	7	If condition not met
	If Z = 0, PC ← PC+e											2	3	12	If condition is met
JP (HL)	PC ← HL	•	•	X	•	X	•	•	•	11 101 001	E9	1	1	4	
JP (IX)	PC ← IX	•	•	X	•	X	•	•	•	11 011 101 11 101 001	DD E9	2	2	8	
JP (IY)	PC ← IY	•	•	X	•	X	•	•	•	11 111 101 11 101 001	FD E9	2	2	8	
DJNZ, e	B ← B-1 If B = 0, continue	•	•	X	•	X	•	•	•	00 010 000 ← e-2 ←	10	2	2	8	If B = 0
	If B ≠ 0, PC ← PC+e											2	3	13	If B ≠ 0

Notes: e represents the extension in the relative addressing mode.

e is a signed two's complement number in the range <126, 129>

e-2 in the op-code provides an effective address of pc+e as PC is incremented by 2 prior to the addition of e.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown, ↕ = flag is affected according to the result of the operation.

CALL AND RETURN GROUP

			CONDITION									
			UN-COND.	CARRY	NON CARRY	ZERO	NON ZERO	PARITY EVEN (overflow)	PARITY ODD (no overflow)	SIGN NEG.	SIGN POS.	REG. B ≠ 0
'CALL'	IMMED. EXT.	nn	CD n n	DC n n	D4 n n	CC n n	C4 n n	EC n n	E4 n n	FC n n	F4 n n	
RETURN 'RET'	REGISTER INDIR.	(SP) (SP+1)	C9	D8	D0	C8	C0	E8	E0	F8	F0	
RETURN FROM INT.'RETI'	REGISTER INDIR.	(SP) (SP+1)	ED 4D									
RETURN FROM NON MASKABLE INT 'RETN'	REGISTER INDIR.	(SP) (SP+1)	ED 45									

NOTE – CERTAIN FLAGS HAVE MORE THAN ONE PURPOSE. REFER TO Z80-CPU TECHNICAL MANUAL FOR DETAILS.

RESTART GROUP

		OP CODE	
CALL ADDRESS	0000_H	C7	'RST 0'
	0008_H	CF	'RST 8'
	0010_H	D7	'RST 16'
	0018_H	DF	'RST 24'
	0020_H	E7	'RST 32'
	0028_H	EF	'RST 40'
	0030_H	F7	'RST 48'
	0038_H	FF	'RST 56'

CALL AND RETURN GROUP

Mnemonic	Symbolic Operation	S	Z		N		P/V	H	C	Op-Code 76	543	210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
CALL nn	(SP-1) ← PC_H (SP-2) ← PC_L PC ← nn	•	•	X	•	X	•	•	•	11 – –	001 n n	101 – –	CD	3	5	17	
CALL cc, nn	If condition cc is false continue, otherwise same as CALL nn	•	•	X	•	X	•	•	•	11 – –	cc n n	100 – –		3 3	3 5	10 17	If cc is false If cc is true
RET	PC_L ← (SP) PC_H ← (SP+1)	•	•	X	•	X	•	•	•	11	001	001	C9	1	3	10	
RET cc	If condition cc is false continue, otherwise same as RET	•	•	X	•	X	•	•	•	11	cc	000		1 1	1 3	5 11	If cc is false If cc is true
RETI	Return from interrupt	•	•	X	•	X	•	•	•	11 01	101 001	101 101	ED 4D	2	4	14	
RETN[1]	Return from non maskable interrupt	•	•	X	•	X	•	•	•	11 01	101 000	101 101	ED 45	2	4	14	
RST p	(SP-1) ← PC_H (SP-2) ← PC_L PC_H ← 0 PC_L ← p	•	•	X	•	X	•	•	•	11	·	111		1	3	11	

cc	Condition	
000	NZ	non zero
001	Z	zero
010	NC	non carry
011	C	carry
100	PO	parity odd
101	PE	parity even
110	P	sign positive
111	M	sign negative

t	p
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

[1] RETN loads $IFF_2 \rightarrow IFF_1$

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
‡ = flag is affected according to the result of the operation.

INPUT GROUP

INPUT DESTINATION			PORT ADDRESS: IMMED. n	PORT ADDRESS: REG. INDIR. (C)
INPUT 'IN'	REG ADDRESSING	A	DB n	ED 78
		B		ED 40
		C		ED 48
		D		ED 50
		E		ED 58
		H		ED 60
		L		ED 68
'INI' – INPUT & Inc HL, Dec B	REG. INDIR	(HL)		ED A2
'INIR' – INP, Inc HL, Dec B, REPEAT IF B≠0				ED B2
'IND' – INPUT & Dec HL, Dec B				ED AA
'INDR' – INPUT, Dec HL Dec B, REPEAT IF B≠0				ED BA

BLOCK INPUT COMMANDS

OUTPUT GROUP

			SOURCE: REGISTER A	B	C	D	E	H	L	REG. IND. (HL)
'OUT'	IMMED.	n	D3 n							
	REG. IND.	(C)	ED 79	ED 41	ED 49	ED 51	ED 59	ED 61	ED 69	
'OUTI' – OUTPUT Inc HL, Dec b	REG. IND.	(C)								ED A3
'OTIR' – OUTPUT, Inc HL, Dec B, REPEAT IF B≠0	REG. IND.	(C)								ED B3
'OUTD' – OUTPUT Dec HL Dec B	REG. IND.	(C)								ED AB
'OTDR' – OUTPUT, Dec HL Dec B, REPEAT IF B ≠ 0	REG. IND.	(C)								ED BB

BLOCK OUTPUT COMMANDS

PORT DESTINATION ADDRESS

INPUT AND OUTPUT GROUP

Mnemonic	Symbolic Operation	Flags S	Z		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
IN A, (n)	A ← (n)	•	•	X	•	X	•	•	•	11 011 011 — n —	DB	2	3	11	n to $A_0 \sim A_7$ Acc to $A_8 \sim A_{15}$
IN r, (C)	r ← (C) if r = 110 only the flags will be affected	↕	↕	X	↕	X	P	0	•	11 101 101 01 r 000	ED	2	3	12	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
INI	(HL) ← (C) B ← B - 1 HL ← HL + 1	X	↕ ①	X	X	X	X	1	•	11 101 101 10 100 010	ED A2	2	4	16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
INIR	(HL) ← (C) B ← B - 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	•	11 101 101 10 110 010	ED B2	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
IND	(HL) ← (C) B ← B - 1 HL ← HL - 1	X	↕ ①	X	X	X	X	1	•	11 101 101 10 101 010	ED AA	2	4	16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
INDR	(HL) ← (C) B ← B - 1 HL ← HL - 1 Repeat until B = 0	X	1	X	X	X	X	1	•	11 101 101 10 111 010	ED BA	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OUT (n), A	(n) ← A	•	•	X	•	X	•	•	•	11 010 011	D3	2	3	11	n to $A_0 \sim A_7$ Acc to $A_8 \sim A_{15}$
OUT (C), r	(C) ← r	•	•	X	•	X	•	•	•	11 101 101 01 r 001	ED	2	3	12	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OUTI	(C) ← (HL) B ← B - 1 HL ← HL + 1	X	↕ ①	X	X	X	X	1	•	11 101 101 10 100 011	ED A3	2	4	16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OTIR	(C) ← (HL) B ← B - 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	•	11 101 101 10 110 011	ED B3	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OUTD	(C) ← (HL) B ← B - 1 HL ← HL - 1	X	↕ ①	X	X	X	X	1	•	11 101 101 10 101 011	ED AB	2	4	16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$
OTDR	(C) ← (HL) B ← B - 1 HL ← HL - 1 Repeat until B = 0	X	1	X	X	X	X	1	•	11 101 101 10 111 011	ED BB	2 2	5 (If B ≠ 0) 4 (If B = 0)	21 16	C to $A_0 \sim A_7$ B to $A_8 \sim A_{15}$

Note: ① If the result of B - 1 is zero the Z flag is set, otherwise it is reset.

Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown,
↕ = flag is affected according to the result of the operation.

Z80 — CPU INTERRUPT STRUCTURE

MASKABLE ($\overline{INT}$)

Mode 0

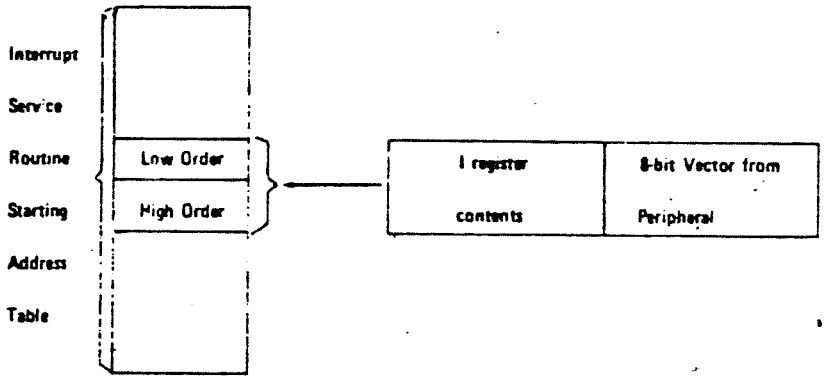
Place instruction onto Data Bus during $\overline{INTA} = \overline{MI} \bullet \overline{IORQ}$ like 8080A

Mode 1

Restart to 38_H or 56_{10} ('RST 56')

Mode 2

Used by Z80 Peripherals



NON MASKABLE ($\overline{NMI}$)

Restart to 66_H or 102_{10}

INTERRUPT ENABLE/DISABLE FLIP-FLOPS

Action	IFF_1	IFF_2	
CPU Reset	0	0	
DI	0	0	
EI	1	1	
LD A, I	•	•	IFF_2 → Parity flag
LD A, R	•	•	IFF_2 → Parity flag
Accept $\overline{NMI}$	0	IFF_1	IFF_1 → IFF_2
RETN	IFF_2	•	IFF_2 → IFF_1
Accept $\overline{INT}$	0	0	
RETI	•	•	

"•" indicates no change

APPENDIX 4

SPECIFICATIONS MT-80Z

CPU:

Z80 Micro Processor, 1.79 mhz clock, crystal controlled

RAM:

2K Supplied - expandable on board to 4K

EProm/Rom:

2K monitor supplied - expandable on board to 8K (with 2K RAM)
4K (with 4K RAM)
6K (with 2K RAM)

Expansion Bus:

STD (Prolog, Mostek) Z80 Bus fully buffered
Full DMA with on/board memory and I/O devices

Parallel I/O:

2 - 8 bit dip switch input ports
1 dedicated and readdressable through the on board solderless breadboard
1 uncommitted - can serve as (2) 4 bit input ports or 2 groups of 4 logic switches
2 - 8 bit LED output ports
1 dedicated and readdressable through the on-board solderless breadboard, also useable as DATA bus monitor
1 uncommitted, can serve as (2) 4 bit output ports or 2 groups of logic indicators

On board parallel I/O expansion with addition of Z-80 Peripheral Interface Adapter and counter/timer chips (user supplied)

Serial I/O:

Audio cassette interface, 165 baud data transfer rate

Data & Function Entry:

36 key keypad, 19 function keys, 16 hexadecimal digit keys and 1 user defined key

Readouts & Displays:

6 - 0.5" high 7 segment LED displays for readout of address and data plus user prompting
2 - sets of 8 LED displays for DATA readout on the two parallel output ports

Hardware Single Step:

Single machine cycle. Run/cycle switch and single step pushbutton patchable to the WAIT line through solderless breadboard interface.

MONITOR:

2K software PROM monitor controls various computer functions such as system initialization, keyboard and display scanning, cassette tape read and write. Register and memory select/modify and display, single step, software break point, memory block transfer and others. All routines are user callable. Automatic software self test.

Audio:

2.25" speaker and audio drive circuits are provided on board for user's special application involving digital tones

Breadboarding Facilities:

All system bus and control lines are buffered and connected to a solderless breadboarding interface socket for easy patching to the on board SK10 breadboarding socket. Approximately 250ma of +5V available for breadboarding experiments

Power Supply:

Unit comes with 1 amp 9VDC unregulated wall adaptor which plugs into on board +5 V regulator; short circuit proof with thermal shutdown protection

+/- VDC : On board regulators; short circuit proof with thermal shutdown protection ****(requires optional +/- 15VDC unregulated adaptor - factory available)

Power Requirements:

Approximately 15 watts

Case:

Polyethelyne blow-molded case with removable locking cover

Size:

11" (27,94 cm) deep x 15" (38,1 cm) wide x 3" (7,62 cm) high

Weight:

Approximately 2 lbs (1 kg)

APPENDIX 5

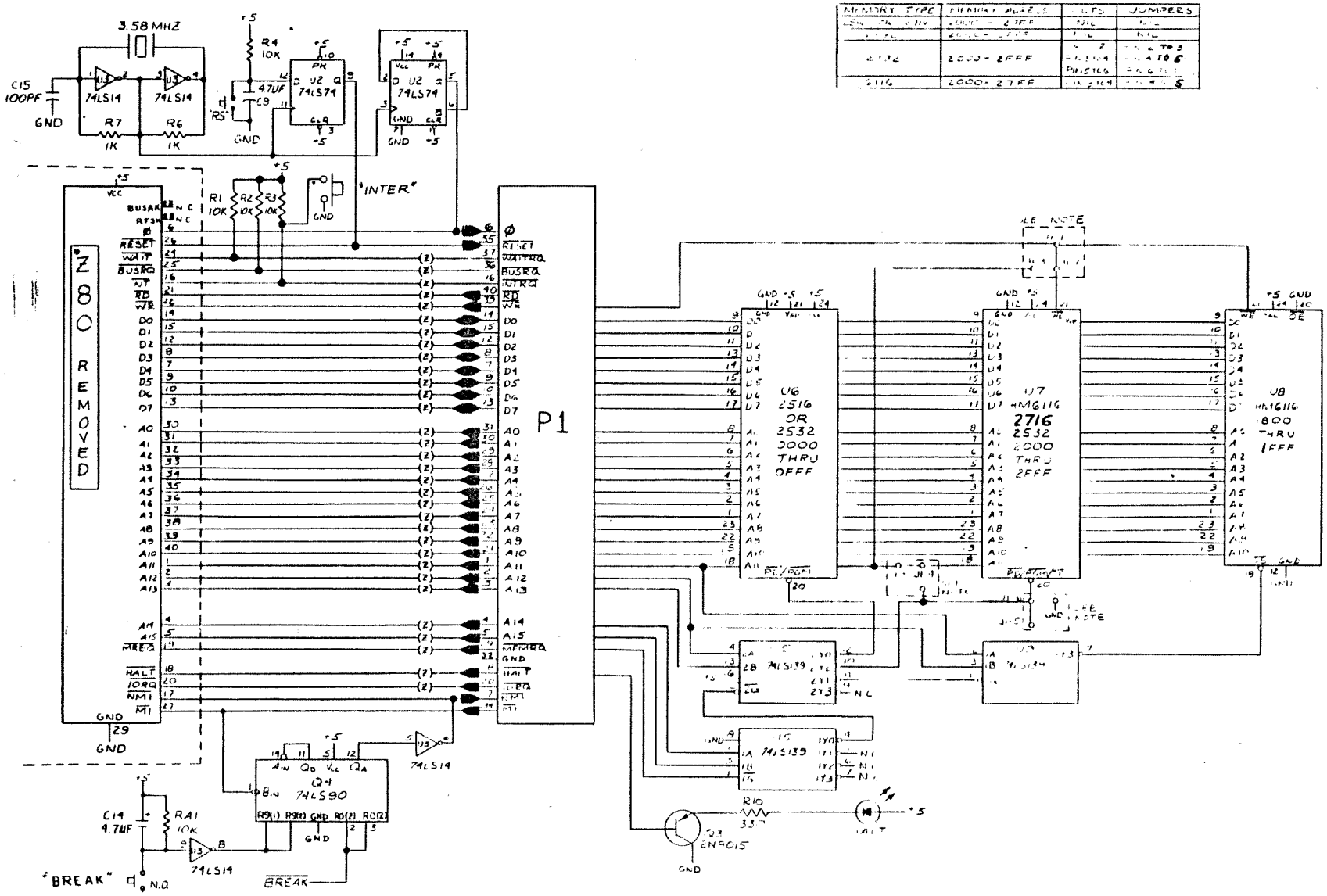
SCHEMATICS

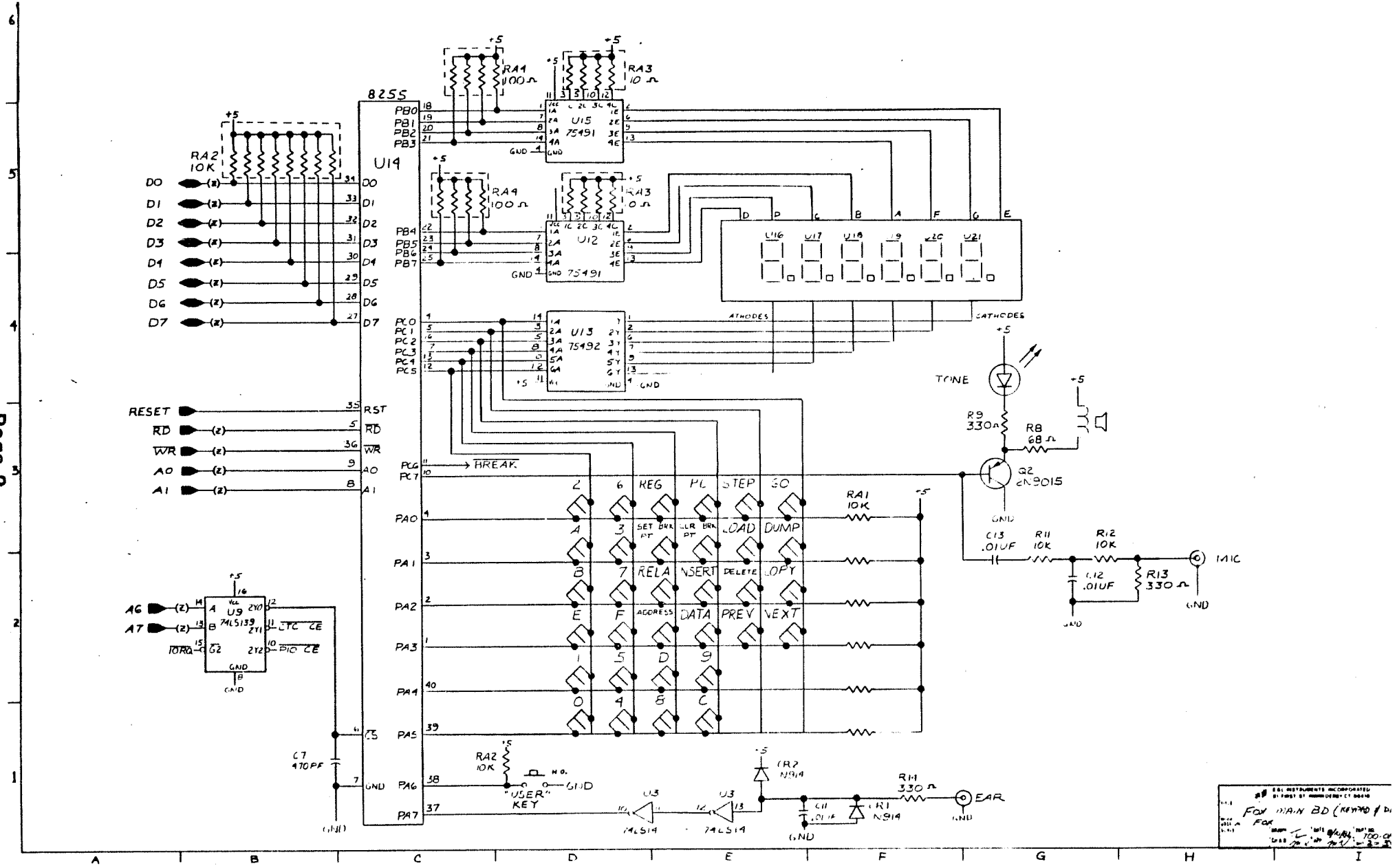
Portions of this information have been reproduced courtesy of:

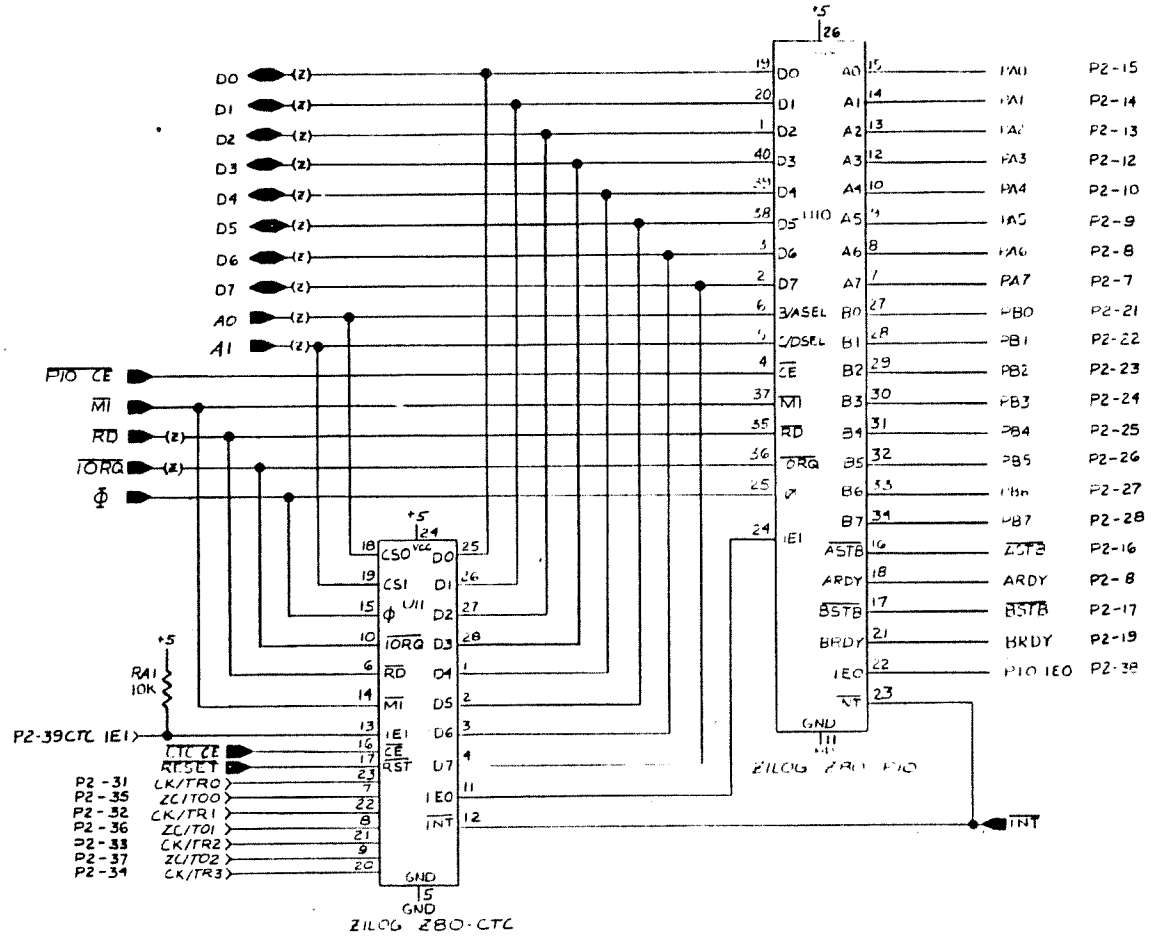
MULTITECH INDUSTRIAL CORPORATION
977 MIN SHEN E. ROAD
TAIPEI 105 TAIWAN
Republic of China

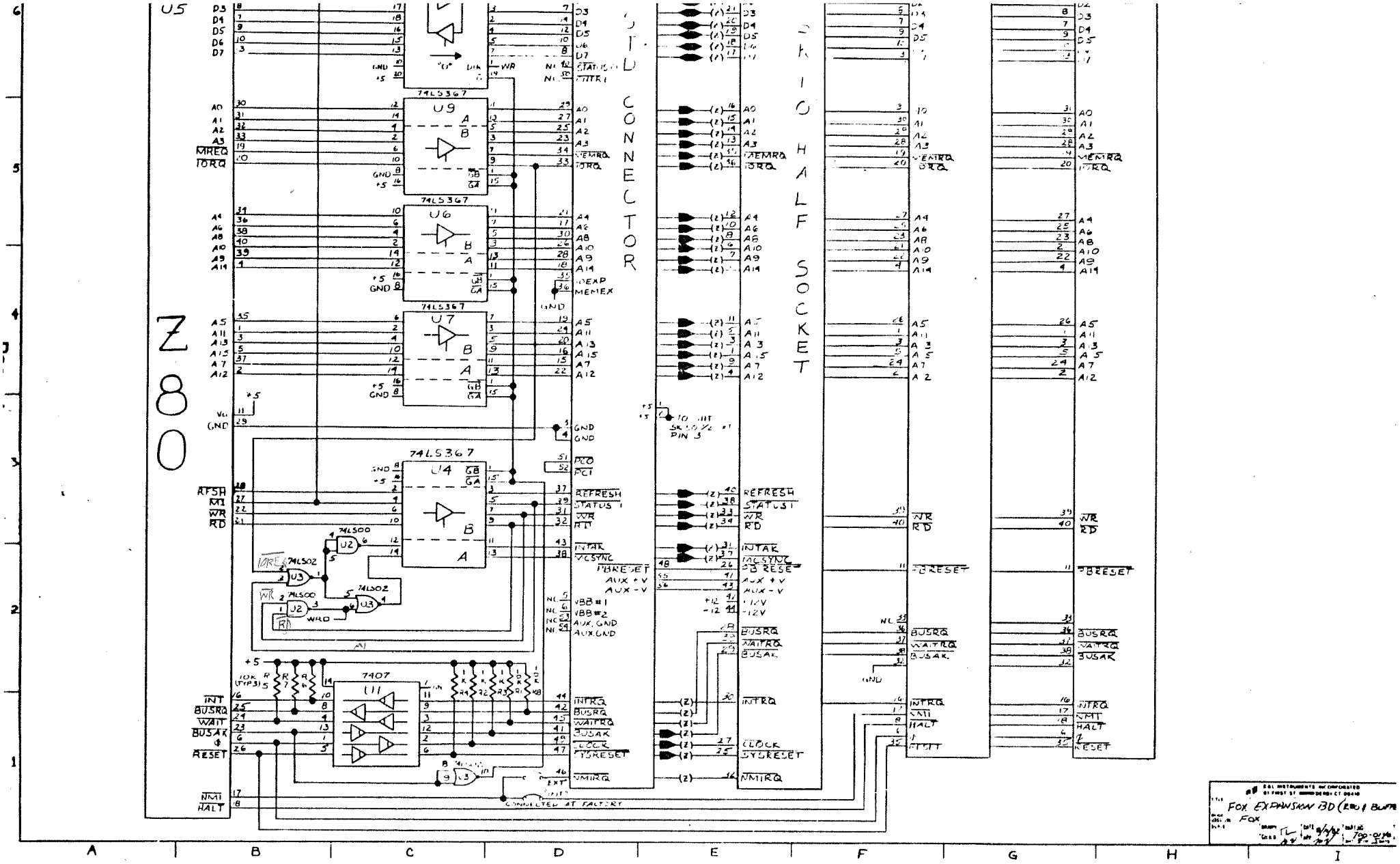
NOTE:
U7 IS OPTIONAL IT MAY BE 2516, 2716, 2532, 2732, OR HM6116
JUMPER 1 IS USED FOR SELECTING CERTAIN MEMORY TYPE

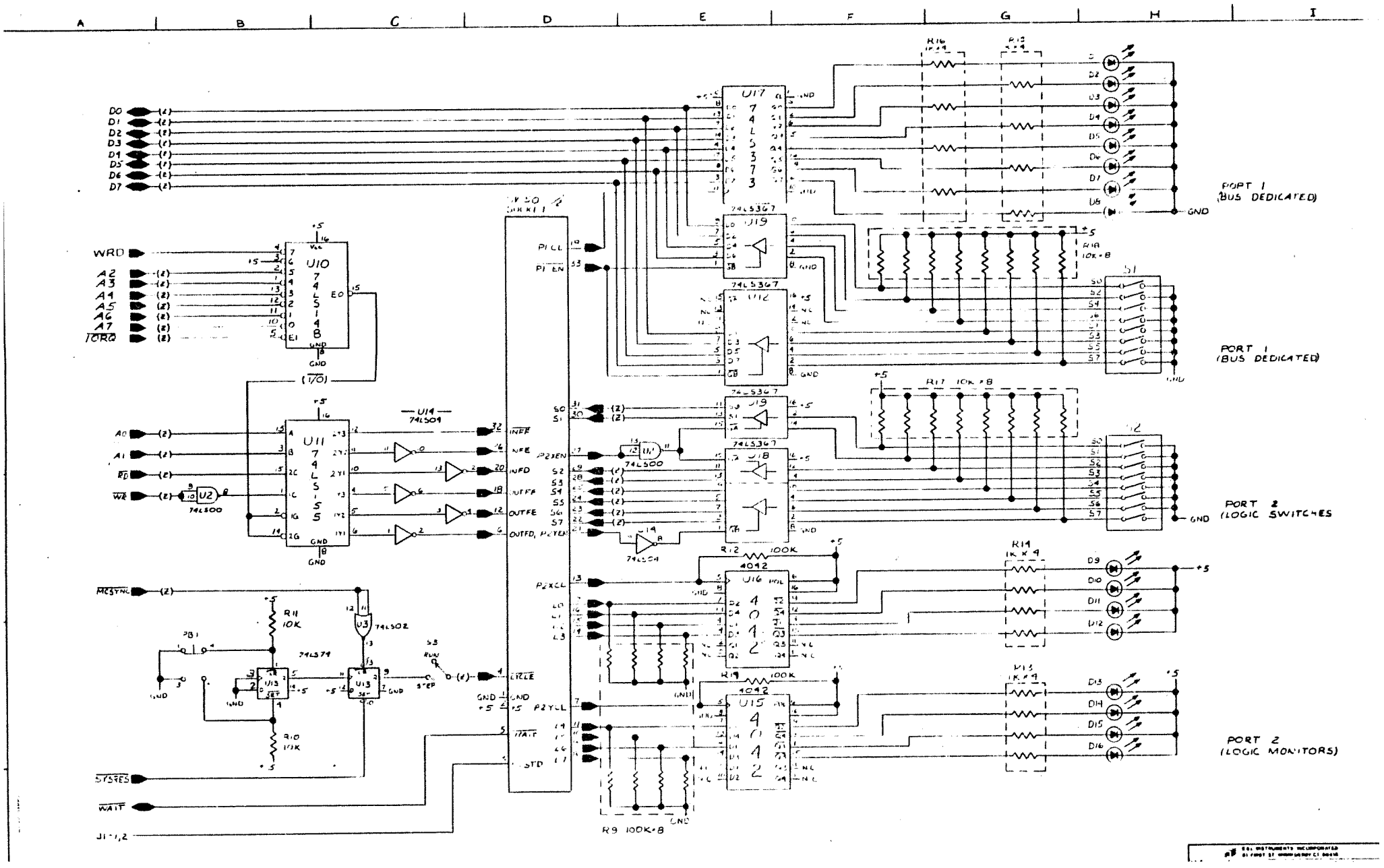
MEMORY TYPE	MEMORY ADDRESS	CTS	JUMPERS
[illegible]	[illegible] - 27FF	[illegible]	[illegible]
[illegible]	[illegible] - [illegible]	[illegible]	[illegible]
2732	2000 - 2FFF	[illegible]	[illegible] TO 3 [illegible] 4 TO 5
		PIN 5	[illegible]
6116	2000 - 27FF	[illegible]	[illegible] 5

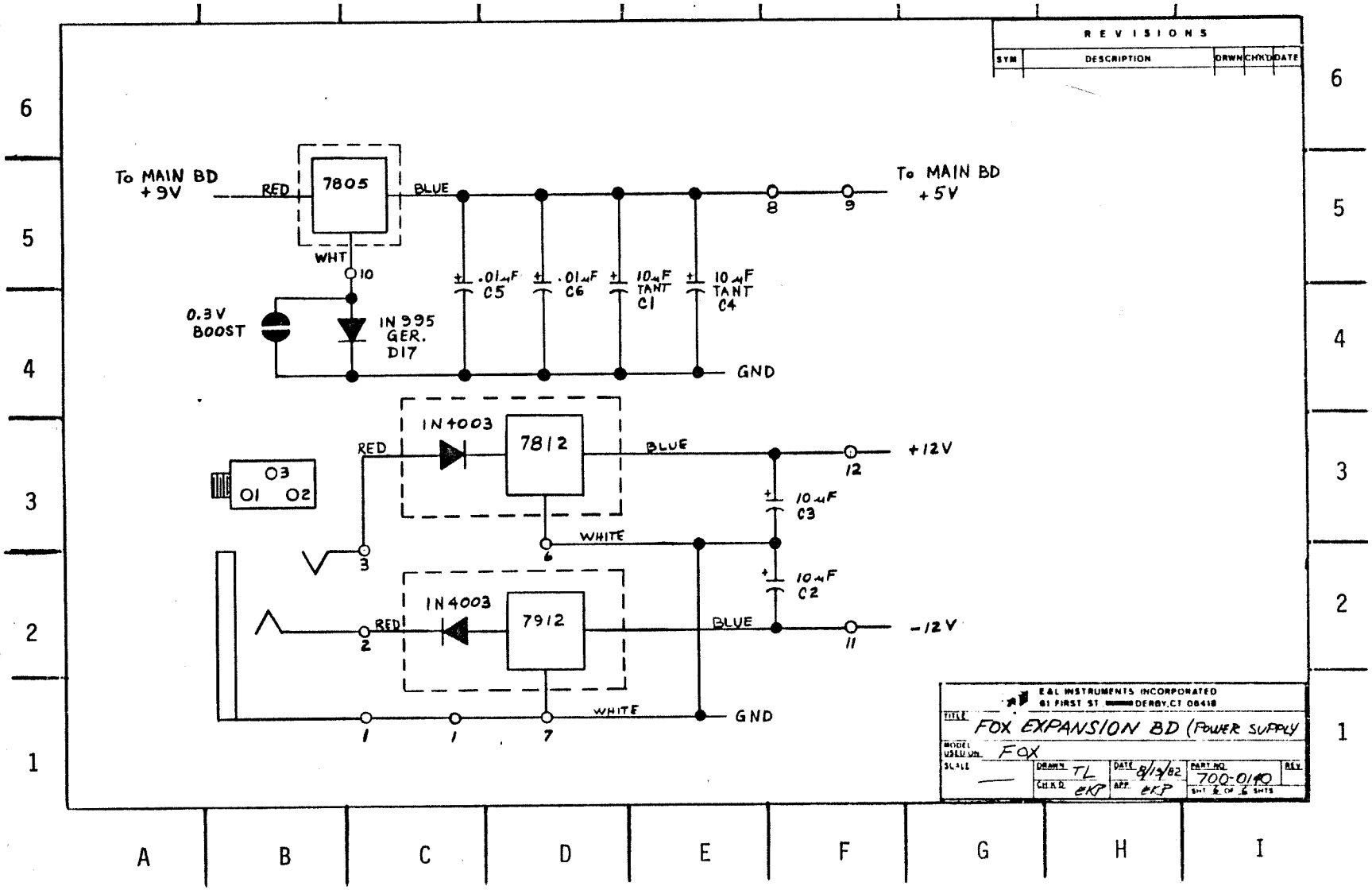



8255
U14
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
D0
D1
D2
D3
D4
D5
D6
D7
RESET
RD
WR
A0
A1
A6
A7
IORQ
CS
GND
RST
BREAK
RA1 100Ω
RA2 10K
RA3 10Ω
RA4 100Ω
U15 75491
U12 75491
U13 75492
U16
U17
U18
U19
U20
U21
ANODES
CATHODES
TONE
R9 330Ω
R8 68Ω
Q2 2N9015
C13 .01UF
R11 10K
R12 10K
C12 .01UF
R13 330Ω
MIC
RA1 10K
2
6
REG
PC
STEP
GO
A
3
SET PT
BRK
CLR PT
BRK
LOAD
DUMP
B
7
RELA
INSERT
DELETE
COPY
E
F
ADDRESS
DATA
PREV
NEXT
1
5
D
9
0
4
8
C
U9 74LS139
CTC CE
PIO CE
C7 470PF
RA2 10K
N.O.
"USER" KEY
U3 74LS14
U3 74LS14
CR2 N914
C11 .01UF
CR1 N914
R14 330Ω
EAR


D0
D1
D2
D3
D4
D5
D6
D7
A0
A1
PIO CE
M1
RD
IORQ
Φ
+5
RA1
10K
P2-39CTC IE1
CTC CE
RESET
P2-31 CK/TR0
P2-35 ZC/TO0
P2-32 CK/TR1
P2-36 ZC/TO1
P2-33 CK/TR2
P2-37 ZC/TO2
P2-34 CK/TR3
CS0
CS1
U11
IORQ
RD
M1
IE1
CE
RST
INT
GND
ZILOG Z80-CTC
B/ASEL
C/DSEL
CE
M1
RD
IORQ
IE1
ASTB
ARDY
BSTB
BRDY
IEO
INT
GND
ZILOG Z80 PIO
PA0 P2-15
PA1 P2-14
PA2 P2-13
PA3 P2-12
PA4 P2-10
PA5 P2-9
PA6 P2-8
PA7 P2-7
PB0 P2-21
PB1 P2-22
PB2 P2-23
PB3 P2-24
PB4 P2-25
PB5 P2-26
PB6 P2-27
PB7 P2-28
ASTB P2-16
ARDY P2-8
BSTB P2-17
BRDY P2-19
PIO IEO P2-38
INT


Z80
U5
U9
U6
U7
U4
U11
74LS367
74LS00
74LS02
7407
STD CONNECTOR
SKIO HALF SOCKET
CONNECTED AT FACTORY
FOX EXPANSION BD (REV) BUFFER
FOX


A B C D E F G H I
D0
D1
D2
D3
D4
D5
D6
D7
WRD
A2
A3
A4
A5
A6
A7
IORQ
U10
74LS138
U17
74LS374
U19
74LS367
U18
U11
74LS155
U14
74LS04
U2
74LS00
U3
74LS02
U13
74LS74
U16
4042
U15
R16
R13
R18
10K*8
R17
R12
100K
R19
R14
1K*4
R15
R11
10K
R10
10K
R9
100K*8
PORT 1
(BUS DEDICATED)
PORT 2
(LOGIC SWITCHES)
PORT 2
(LOGIC MONITORS)
MCSYNC
SYSRES
WAIT
J1-1,2
PB1
S3
RUN
STEP
GND
+5


REVISIONS
SYM
DESCRIPTION
DRWN
CHKD
DATE
To MAIN BD
+9V
RED
7805
BLUE
WHT
10
0.3V
BOOST
IN995
GER.
D17
.01μF
C5
.01μF
C6
10μF
TANT
C1
10μF
TANT
C4
8
9
To MAIN BD
+5V
GND
IN4003
7812
RED
BLUE
12
+12V
10μF
C3
WHITE
6
3
10μF
C2
IN4003
7912
RED
2
BLUE
11
-12V
WHITE
GND
1
1
7
E&L INSTRUMENTS INCORPORATED
61 FIRST ST DERBY, CT 06418
TITLE
FOX EXPANSION BD (POWER SUPPLY
MODEL USED ON
FOX
SCALE
DRAWN TL
DATE 8/17/82
PART NO. 700-0140
REV.
CHKD EKP
APP. EKP
SHT 1 OF 1 SHTS

APPENDIX 6

MEMORY EXPANSION - External

The MT-80Z memory system can be expanded externally to 64k by using STD memory cards. In general, expanded external memory can be prom, static ram or dynamic ram. However, there are some things to consider before attempting to add memory on the MT-80Z.

1) Make sure that you have enough power to supply the external memory : There is approximately 250 ma of breadboarding power available from the MT-80Z power supply

2) Make sure that the external memory map isn't in conflict with the internal memory map

3) There are two types of dynamic ram cards available these days: those that depend upon the Z80 for refresh operation and those that don't. Both will work provided that you don't exceed their speed criteria relative to the clock rate of the MT-80Z with one exception. Dynamic ram cards that do depend on the Z80 refresh and address lines to refresh their ram will LOOSE their DATA if you attempt to use the hardware single step option of the MT-80Z. Inevitably the refresh line will be held high during single step operation and the address lines will not be chinging which precludes using this type of dynamic ram card. Only those cards that have an on board refresh signal generator will work during single step operation

Portions of this information have been reproduced courtesy of:

MULTITECH INDUSTRIAL CORPORATION
977 MIN SHEN E. ROAD
TAIPEI 105 TAIWAN
Republic of China

MEMORY EXPANSION/CONVERSION - Internal

The MT-80Z on board memory can be expanded to several configurations. Some of these configurations require that the PC board be removed from the case to perform a cut and patch modification. The various configurations are listed below.

2532 EXPANSION

U6 - Expands from 2K prom to 4k prom
a)Replace 2716 with 2532
b)No cut and patch necessary

U7 - Expands from 2k prom to 4k prom
a)Replace 2716 prom with 2532 prom
b)No cut and patch necessary

U8 - Not expandable 2532

2732 EXPANSION

U6 - Not expandable to type 2732

U7 - Expands from 2k prom to 4k prom
a)Replace 2716 with 2732
b)Cut J1-1,2
Cut J1-3,4
Cut J1-5,6
c)Patch J1-2,3
Patch J1-4,5
Patch J1-6,7

U8 Not expandable to type 2732

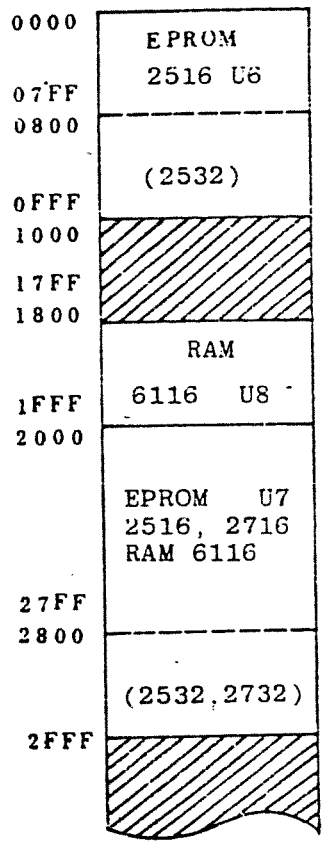
PROM TO RAM CONVERSION

U6 Not convertable to type 6116

U7 - Converts from 2k prom to 2k ram
a)Replace prom with 6116
b)Cut J1-3,4
c)Patch J1-4,5

U8 No conversion necessary

MEMORY MAP



MT-80Z DISPLAY

The display of the MT-80Z is composed of six individual seven segment displays. The information presented on these displays is multiplexed by the software of the monitor program via output port U14 (see sheet 2 of schematics).

The individual displays are controlled by 14 signals (PC0 thru PC5) and (PB0 thru PB7). Six of these signals (PC0 thru PC5) determine which one of the six displays will be active while the remaining 8 signals, (PB0 thru PB7), define the individual segments and decimal points which will be illuminated.

Users can present their own information on the displays as follows:

The information patterns are defined by the user in a "buffer" zone in memory and then the user calls a subroutine in the monitor program. The monitor subroutine will distribute the user information on the displays and return.

Because the displays are multiplexed by the software it is necessary for the user to write a loop program which continues to call the monitor subroutine in order to maintain the information on the displays. If the user dosen't maintain the loop the information on the displays will collaspse.

The location of the display buffer is somewhat arbitrary and can be defined by the user by setting the IX register of the Z80 to that section of memory where the programmer (user) wants to allocate space for the message. The following example demonstrates how to display messages using the monitor subroutines.

HOW TO DISPLAY A MESSAGE

-General Method -

1) Pick a convenient point in memory as a "buffer" for your message
2) Using the following example and table as a guide, store the necessary codes for your message in the "buffer" zone
3) Write a loop which calls the monitor subroutine continuously until some detectable event occurs (i.e. interrupt, keystroke etc..)
4) Upon detection of the event, take the appropriate action but bear in mind that the DISPLAYS WILL COLLAPSE once you leave the monitor subroutine
5) There are two subroutines which can be used to handle the display SCAN and SCAN1. The differences in these routines are explained in the following pages

KEYBOARD AND DISPLAY

The following information is provided to help you:

1) Understand how the MT-80Z display works

2) Understand how the MT-80Z interprets the keyboard

3) Understand how to use the monitor subroutines to control the display and interpret the keyboard for your own purposes

Portions of this information have been reproduced courtesy of:

MULTITECH INDUSTRIAL CORPORATION
977 MIN SHEN E. ROAD
TAIPEI 105 TAIWAN
Republic of China

SCAN1

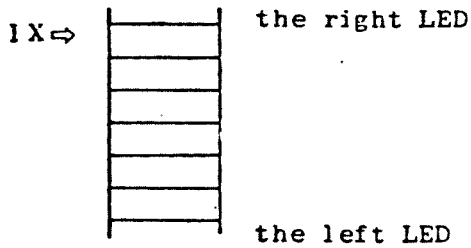
[Address]: 0624

Function]: Scan keyboard and the display 1 cycle from right to left. Execution time is about 10ms (9.97ms exactly).-

[Input]: IX points to the display buffer.

[Output]: (1) If no key-in, then carry flag = 1.
(2) If key-in, carry flag = 0 and the position-code of the key is stored in register A.

[Supplement]: (1) 6 bytes are required for 6 LED's.
(2) IX points to the rightmost LED, IX+5 points to the left most LED.



(3) See Fig. 3-11-4 for the relation between each bit and the seven segments.

SCAN

[Address]: 05FE

[Function]: Similar to SCAN1 except:

(1) SCAN1 scans one cycle, but SCAN will scan till a new key-in.
(2) SCAN1 returns the position while SCAN returns the internal code of the key pressed.

[Input]: IX points to the display buffer.

[Output]: Register A contains the internal code of the key pressed.

[Register]: Destroy AF,B, HL, AF', BC', DE'.

EXAMPLE : Display HELPUS , HALT when Step is pressed.

```
                          DISPLAY 'HELPUS' UNTIL KEY-STEP PUSHED:
1800                      ORG     1800H
1800  DD212018            LD      IX,HELP
1804  CDFE05              CALL    SCAN
1807  FE13                CP      13H          ;KEY-STEP
1809  20F9                JR      NZ,DISP
180B  76                  HALT
                          ;
1820                      ORG     1820H
1820  AE                  DEFB    0AEH         ; 'S'
1821  B5                  DEFB    0B5H         ; 'U'
1822  1F                  DEFB    01FH         ; 'P'
1823  85                  DEFB    085H         ; 'L'
1824  8F                  DEFB    08FH         ; 'E'
1825  37                  DEFB    037H         ; 'H'
                          ;
                          EQU     05FEH
                          END
```

Details of the display buffer are given below:

7	6	5	4	3	2	1	0
d	p	c	b	a	f	g	e

Position	Display Format	Segment of Illumination	d p c b a f g e	Data	Addr
Right ↕ Left	S	a,c,d,f,g,	1 0 1 0 1 1 1 0	AE	1820
	U	b,c,d,e,f,	1 0 1 1 0 1 0 1	B5	1821
	P	a,b,e,f,g,	0 0 0 1 1 1 1 1	1F	1822
	L	d,e,f,	1 0 0 0 0 1 0 1	85	1823
	E	a,d,e,f,g,	1 0 0 0 1 1 1 1	8F	1824
	H	b,c,e,f,g,	0 0 1 1 0 1 1 1	37	1825

CODE	BD	30	9B	BA	36	AE	AF	3B	BF	BE	3F	A7	BD	B3
DATA	0	1	2	3	4	5	6	7	8	9	A	B	C	D
DISP	0	1	2	3	4	5	6	7	8	9	A	b	C	d

CODE	8F	0F	AD	37	89	81	97	85	2B	23	A3	1F	3E	03
DATA	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
DISP	E	F	G	H	I	J	K	L	M	n	o	P	q	r

CODE	A6	B7	B5	B7	A9	07	B6	BA	B3	A2	32	02	C0	00
DATA	S	T	U	V	W	X	Y	Z	(	)	+	−	,	
DISP	S	t	U	V	W	X	Y	Z	(	)	+	−	,	

MT-80Z KEYBOARD

The MT-80Z keyboard is composed of 36 keys arranged in a 6 by 6 matrix. The matrix is "scanned" by the software of the monitor routines via I/O port U14 (see sheet 2 of schematicss)

The individual keys are software driven by I/O port U14 bits (PC0 thru PC5). Closure is detected through software by analyzing I/O port U14 bits (PA0 thru PA5). Debouncing is also accomplished through software. And finally, an identification code is assigned to each key by software, and placed in the "A" register of the Z80 for appropriate response by other programs. Users can interpret the keyboard for their own purposes by CALLing one of the two keyboard subroutines SCAN1 or SCAN . An explanation of the two routines and examples of their use is given in the following pages.

Please notice that both routines SCAN1 and SCAN will effect the displays !

SCAN1

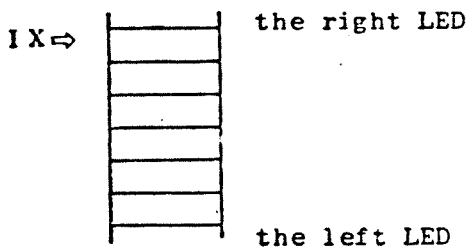
[Address]: 0624

Function]: Scan keyboard and the display 1 cycle from right to left. Execution time is about 10ms (9.97ms exactly).

[Input]: IX points to the display buffer.

[Output]: (1) If no key-in, then carry flag = 1.
(2) If key-in, carry flag = 0 and the position-code of the key is stored in register A.

[Supplement]: (1) 6 bytes are required for 6 LED's.
(2) IX points to the rightmost LED, IX+5 points to the left most LED.



SCAN

[Address]: 05FE

[Function]: Similar to SCAN1 except:

(1) SCAN1 scans one cycle, but SCAN will scan till a new key-in.
(2) SCAN1 returns the position while SCAN returns the internal code of the key pressed.

[Input]: IX points to the display buffer.

[Output]: Register A contains the internal code of the key pressed.

[Register]: Destroy AF,B, HL, AF', BC', DE'.

EXAMPLE : Display the key code of the key pressed.

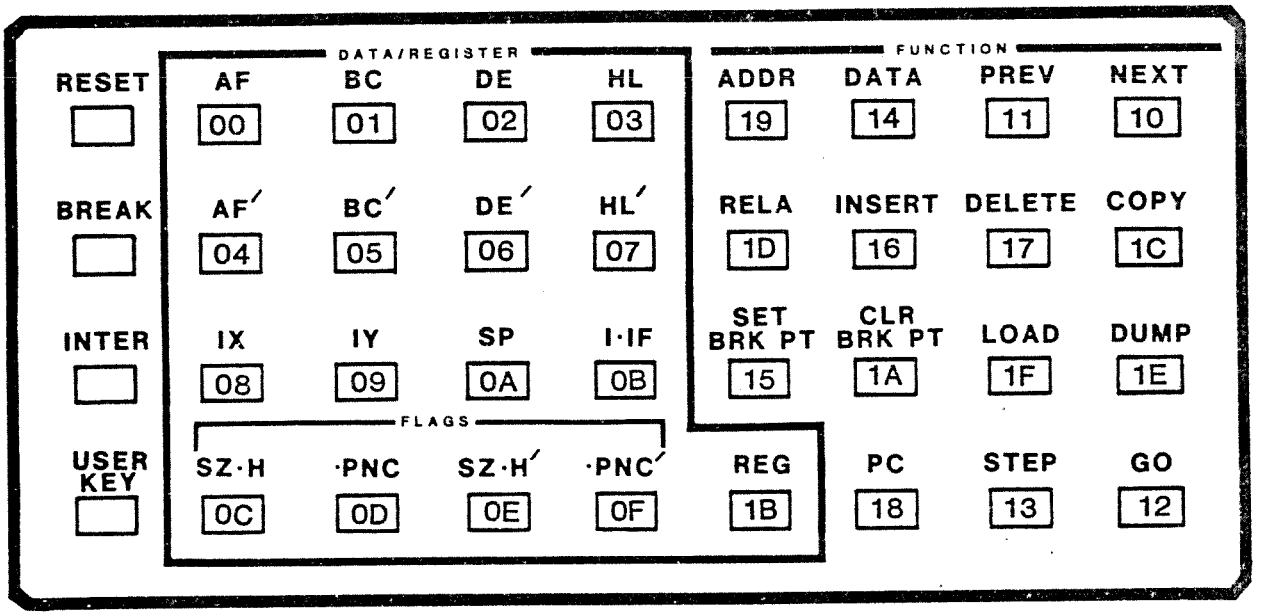
```
                        1    ;DISPLAY INTERNAL CODE
1800                    2              ORG       1800H
1800  DD210019          3              LD        IX,OUTBF
1804  CDFE05            4  LOOP        CALL      SCAN
1807  210019            5              LD        HL,OUTBF
180A  CD7806            6              CALL      HEX7SG
180D  18F5              7              JR        LOOP
                        8  ;
1900                    9              ORG       1900H
1900  00                10 OUTBF       DEFB      0
1901  00                11             DEFB      0
1902  00                12             DEFB      0
1903  00                13             DEFB      0
1904  00                14             DEFB      0
1905  00                15             DEFB      0
                        16 ;
                        17 SCAN        EQU       05FEH
                        18 HEX7SG      QU        0678H
                        19             END
```

When a key is pressed, the internal code of it is displayed on the data filed. The user may compare it with Fig. 2-11-5.

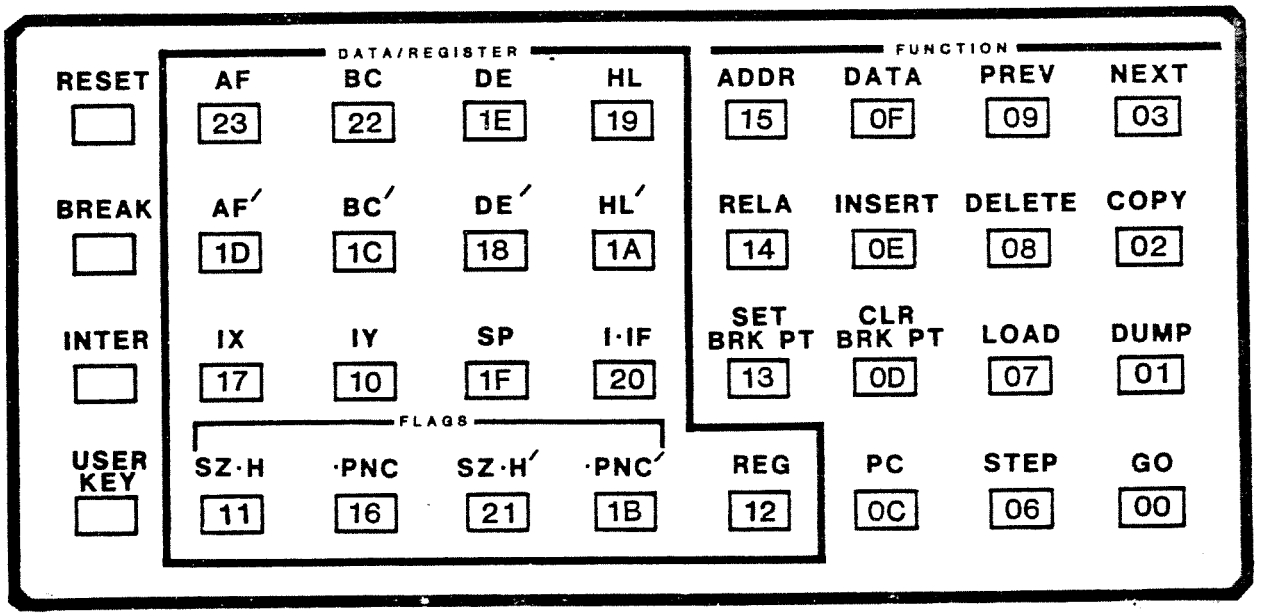
If you want to display the position code of the keys, you may change the program as follow:

```
                        1    ;DISPLAY POSITION CODE
1800                    2              ORG       1800H
1800  DD210019          3              LD        IX,OUTBF
1800  CD2406            4  LOOP        CALL      SCAN1
1807  38FB              5              JR        C,LOOP
1809  210019            6              LD        HL,OUTBF
180C  CD7806            7              CALL      HEX7SG
180F  18F3              8              JR        LOOP
                        9
```

Position-Code (CALL SCAN1):



Internal-Code (CALL SCAN):

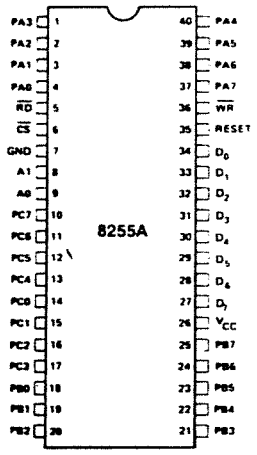


8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

- MCS-85™ Compatible 8255A-5
- 24 Programmable I/O Pins
- Completely TTL Compatible
- Fully Compatible with Intel® Microprocessor Families
- Improved Timing Characteristics
- Direct Bit Set/Reset Capability Easing Control Application Interface
- 40-Pin Dual In-Line Package
- Reduces System Package Count
- Improved DC Driving Capability

The Intel® 8255A is a general purpose programmable I/O device designed for use with Intel® microprocessors. It has 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. In the first mode (MODE 0), each group of 12 I/O pins may be programmed in sets of 4 to be input or output. In MODE 1, the second mode, each group may be programmed to have 8 lines of input or output. Of the remaining 4 pins, 3 are used for handshaking and interrupt control signals. The third mode of operation (MODE 2) is a bidirectional bus mode which uses 8 lines for a bidirectional bus, and 5 lines, borrowing one from the other group, for handshaking.

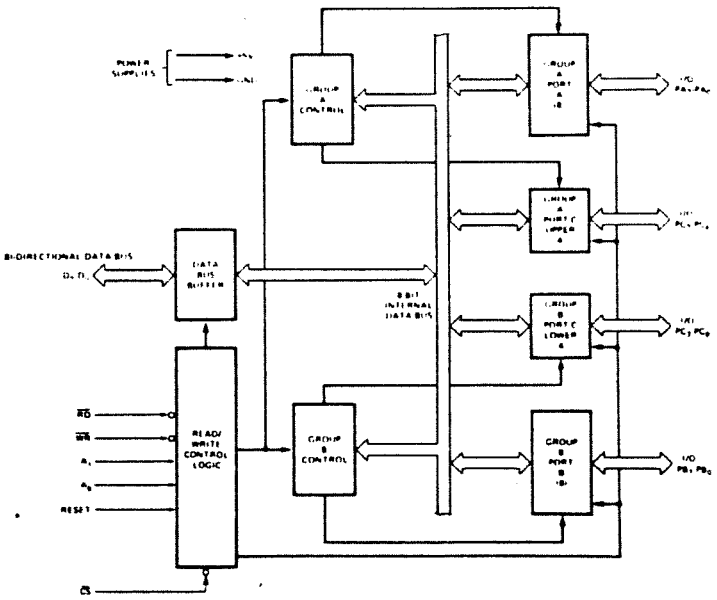
PIN CONFIGURATION



PIN NAMES

D_7–D_0	DATA BUS (BI-DIRECTIONAL)
RESET	RESET INPUT
$\overline{CS}$	CHIP SELECT
$\overline{RD}$	READ INPUT
$\overline{WR}$	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
V_{CC}	+5 VOLTS
GND	0 VOLTS

8255A BLOCK DIAGRAM



8255A FUNCTIONAL DESCRIPTION

General

The 8255A is a programmable peripheral interface (PPI) device designed for use in Intel® microcomputer systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the microcomputer system bus. The functional configuration of the 8255A is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

Data Bus Buffer

This 3-state bidirectional 8-bit buffer is used to interface the 8255A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.

Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

($\overline{CS}$)

Chip Select. A "low" on this input pin enables the communiction between the 8255A and the CPU.

($\overline{RD}$)

Read. A "low" on this input pin enables the 8255A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 8255A.

($\overline{WR}$)

Write. A "low" on this input pin enables the CPU to write data or control words into the 8255A.

(A_0 and A_1)

Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word registers. They are normally connected to the least significant bits of the address bus (A_0 and A_1).

8255A BASIC OPERATION

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0	0	0	1	0	PORT A → DATA BUS
0	1	0	1	0	PORT B → DATA BUS
1	0	0	1	0	PORT C → DATA BUS
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	DATA BUS → PORT A
0	1	1	0	0	DATA BUS → PORT B
1	0	1	0	0	DATA BUS → PORT C
1	1	1	0	0	DATA BUS → CONTROL
					DISABLE FUNCTION
X	X	X	X	1	DATA BUS → 3-STATE
1	1	0	1	0	ILLEGAL CONDITION
X	X	1	1	0	DATA BUS → 3-STATE

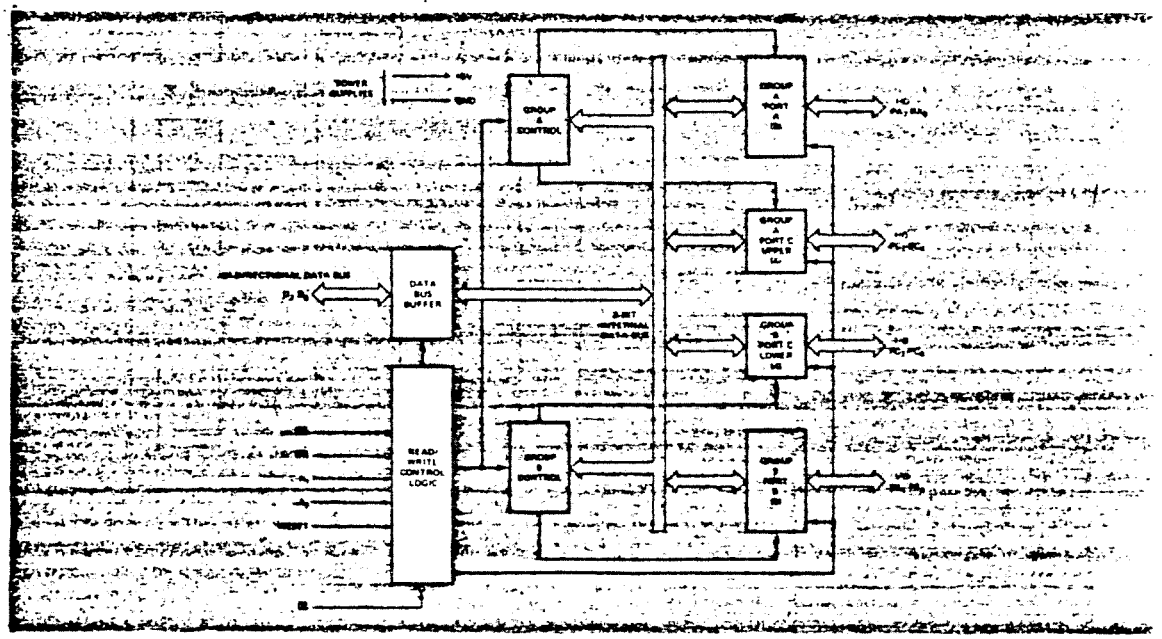


Figure 1. 8255A Block Diagram Showing Data Bus Buffer and Read/Write Control Logic Functions

8255A OPERATIONAL DESCRIPTION

Mode Selection

There are three basic modes of operation that can be selected by the system software:

Mode 0 — Basic Input/Output
Mode 1 — Strobed Input/Output
Mode 2 — Bi-Directional Bus

When the reset input goes "high" all ports will be set to the input mode (i.e., all 24 lines will be in the high impedance state). After the reset is removed the 8255A can remain in the input mode with no additional initialization required. During the execution of the system program any of the other modes may be selected using a single output instruction. This allows a single 8255A to service a variety of peripheral devices with a simple software maintenance routine.

The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance; Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.

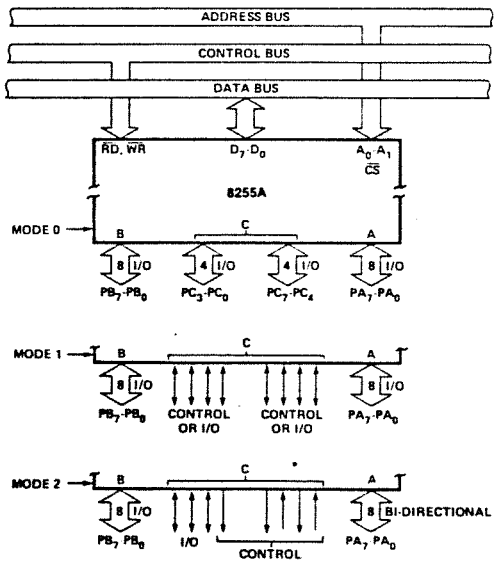


Figure 3. Basic Mode Definitions and Bus Interface

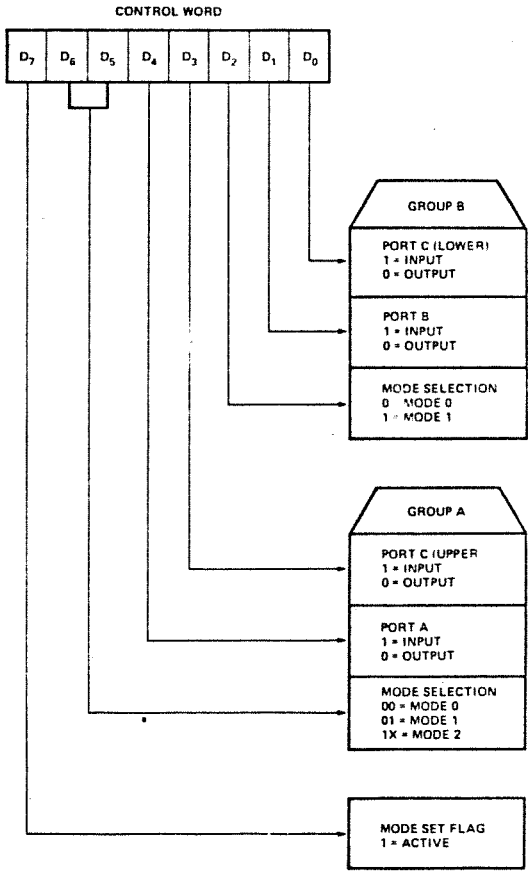


Figure 4. Mode Definition Format

The mode definitions and possible mode combinations may seem confusing at first but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 8255A has taken into account things such as efficient PC board layout, control signal definition vs PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

Single Bit Set/Reset Feature

Any of the eight bits of Port C can be Set or Reset using a single OUTput instruction. This feature reduces software requirements in Control-based applications.

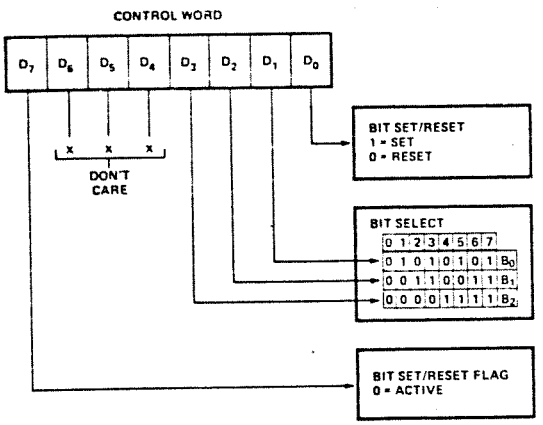


Figure 5. Bit Set/Reset Format

When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.

Interrupt Control Functions

When the 8255A is programmed to operate in mode 1 or mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the bit set/reset function of port C.

This function allows the Programmer to disallow or allow a specific I/O device to interrupt the CPU without affecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET) — INTE is SET — Interrupt enable
(BIT-RESET) — INTE is RESET — Interrupt disable

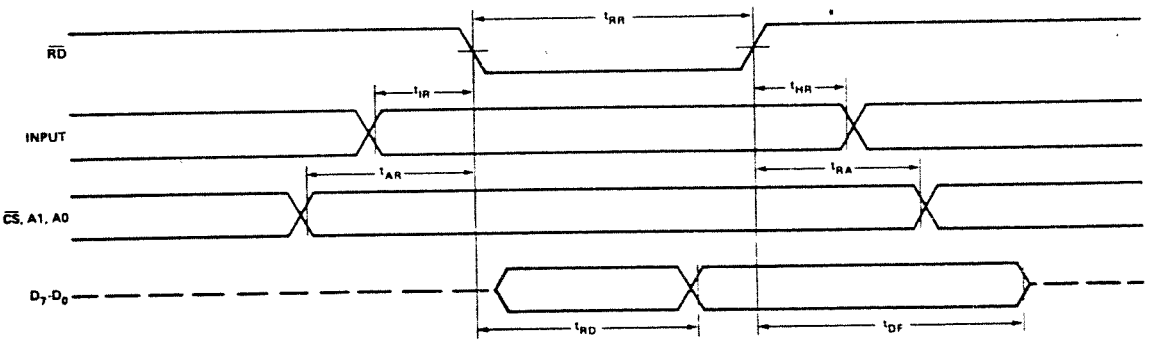
Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

Operating Modes

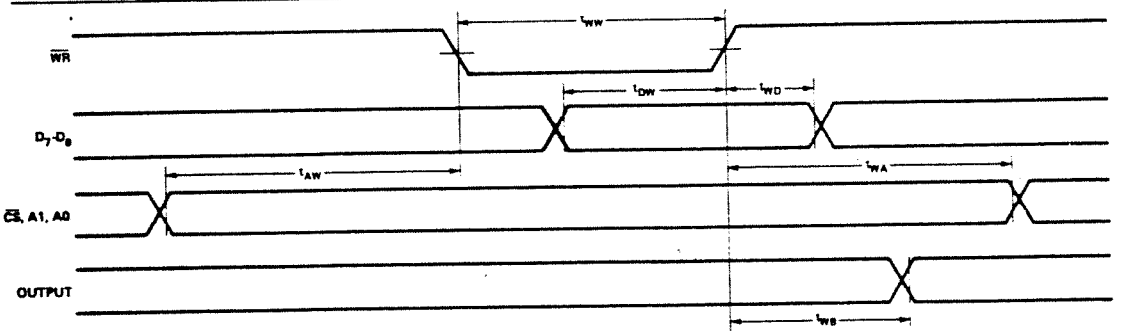
MODE 0 (Basic Input/Output). This functional configuration provides simple input and output operations for each of the three ports. No "handshaking" is required, data is simply written to or read from a specified port.

Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports.
- Any port can be input or output.
- Outputs are latched.
- Inputs are not latched.
- 16 different Input/Output configurations are possible in this Mode.



MODE 0 (Basic Input)

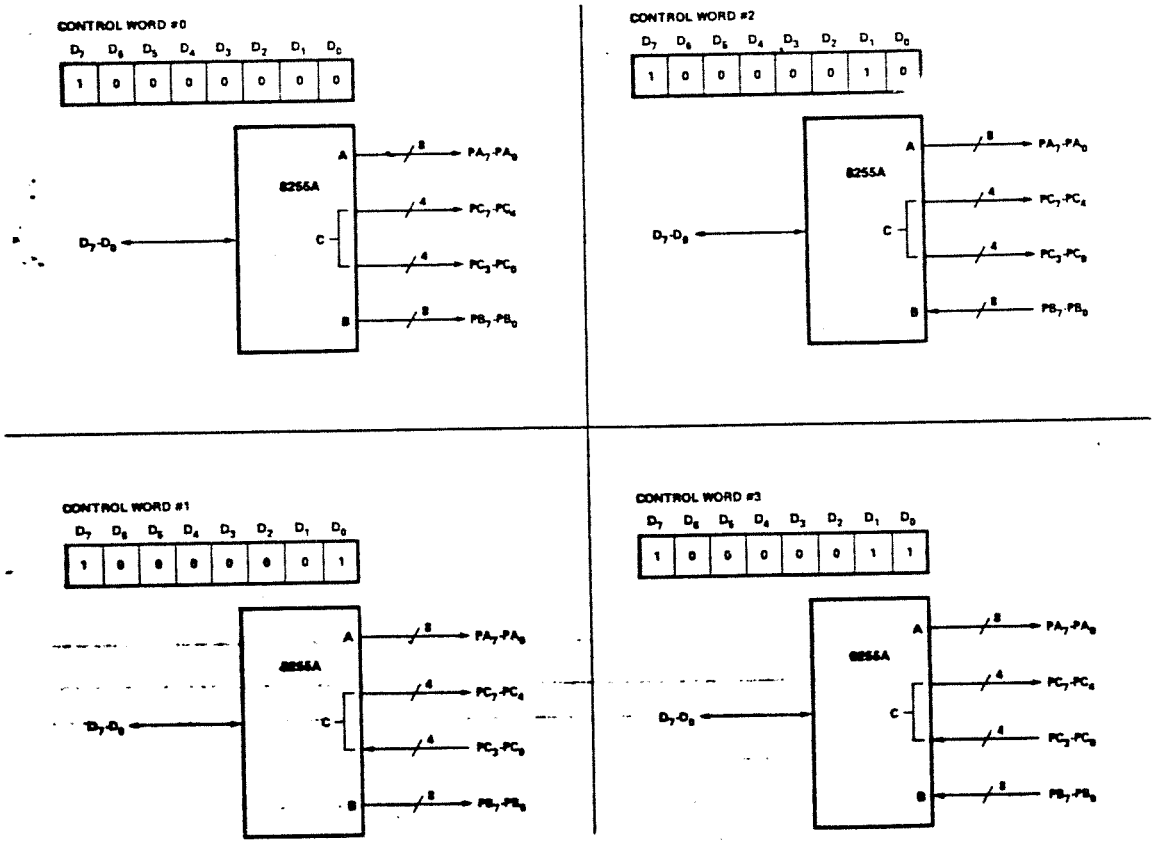


MODE 0 (Basic Output)

MODE 0 Port Definition

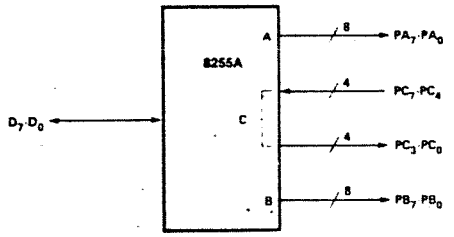
A		B		GROUP A			GROUP B	
D_4	D_3	D_1	D_0	PORT A	PORT C (UPPER)	#	PORT B	PORT C (LOWER)
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	INPUT
0	1	1	0	OUTPUT	INPUT	6	INPUT	OUTPUT
0	1	1	1	OUTPUT	INPUT	7	INPUT	INPUT
1	0	0	0	INPUT	OUTPUT	8	OUTPUT	OUTPUT
1	0	0	1	INPUT	OUTPUT	9	OUTPUT	INPUT
1	0	1	0	INPUT	OUTPUT	10	INPUT	OUTPUT
1	0	1	1	INPUT	OUTPUT	11	INPUT	INPUT
1	1	0	0	INPUT	INPUT	12	OUTPUT	OUTPUT
1	1	0	1	INPUT	INPUT	13	OUTPUT	INPUT
1	1	1	0	INPUT	INPUT	14	INPUT	OUTPUT
1	1	1	1	INPUT	INPUT	15	INPUT	INPUT

MODE 0 Configurations



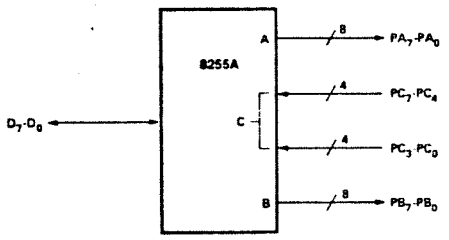
CONTROL WORD #4

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	0



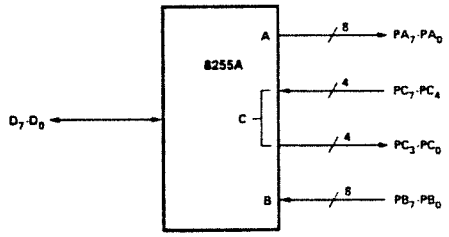
CONTROL WORD #5

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	1



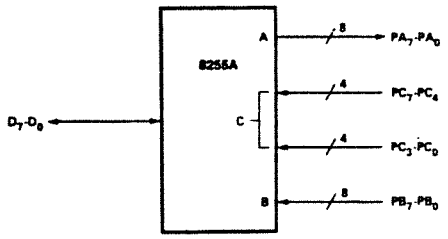
CONTROL WORD #6

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	0



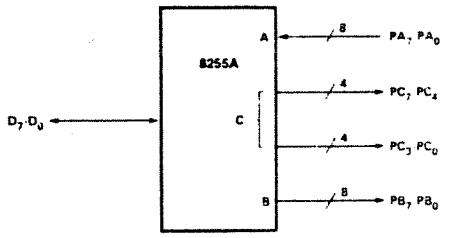
CONTROL WORD #7

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	1



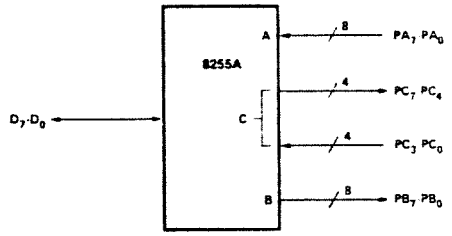
CONTROL WORD #8

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	0



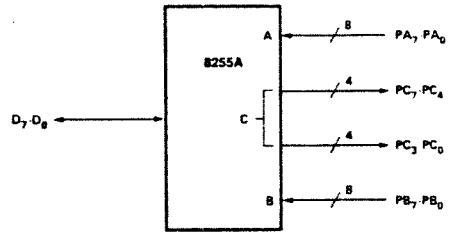
CONTROL WORD #9

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	1



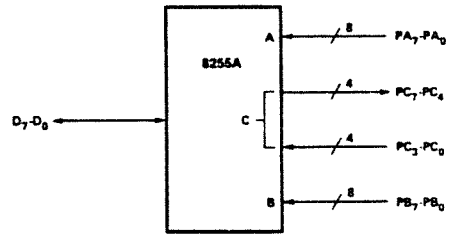
CONTROL WORD #10

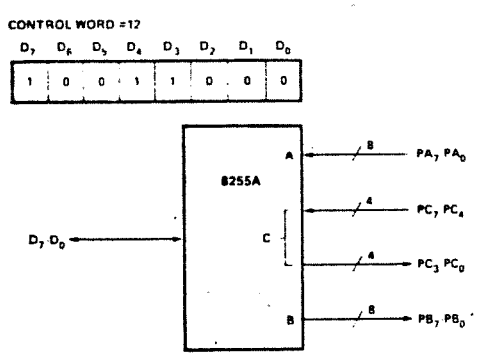
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	0

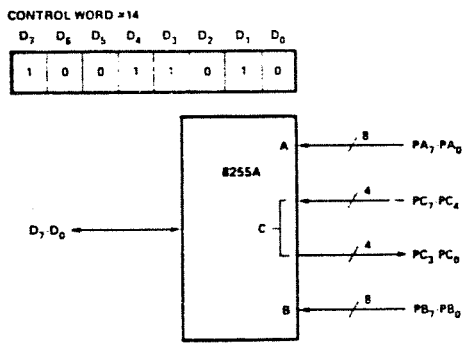


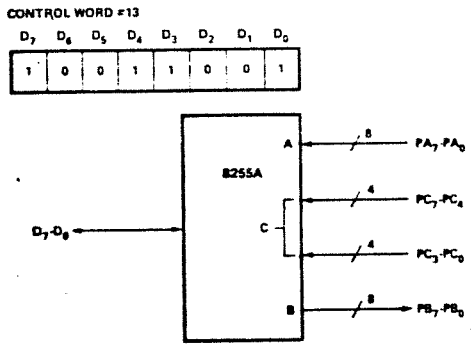
CONTROL WORD #11

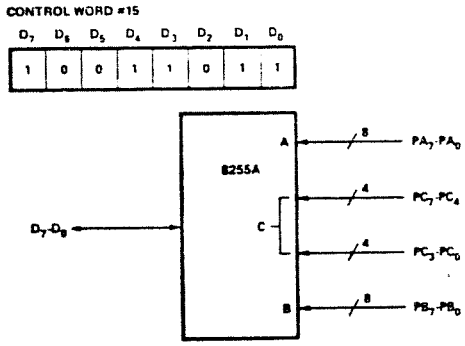
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	1











Operating Modes

MODE 1 (Strobed Input/Output). This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In mode 1, port A and Port B use the lines on port C to generate or accept these "handshaking" signals.

Mode 1 Basic Functional Definitions:

- Two Groups (Group A and Group B)
- Each group contains one 8-bit data port and one 4-bit control/data port.
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit data port.

Output Control Signal Definition

$\overline{OBF}$ (Output Buffer Full F/F). The OBF output will go "low" to indicate that the CPU has written data out to the specified port. The OBF F/F will be set by the rising edge of the WR input and reset by ACK Input being low.

$\overline{ACK}$ (Acknowledge Input). A "low" on this input informs the 8255A that the data from port A or port B has been accepted. In essence, a response from the peripheral device indicating that it has received the data output by the CPU.

INTR (Interrupt Request). A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set when ACK is a "one", OBF is a "one" and INTE is a "one". It is reset by the falling edge of WR.

INTE A

Controlled by bit set/reset of PC_6.

INTE B

Controlled by bit set/reset of PC_2.

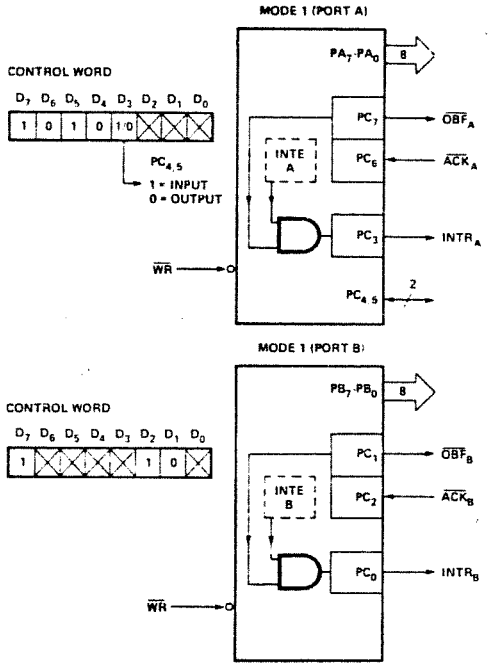


Figure 8. MODE 1 Output

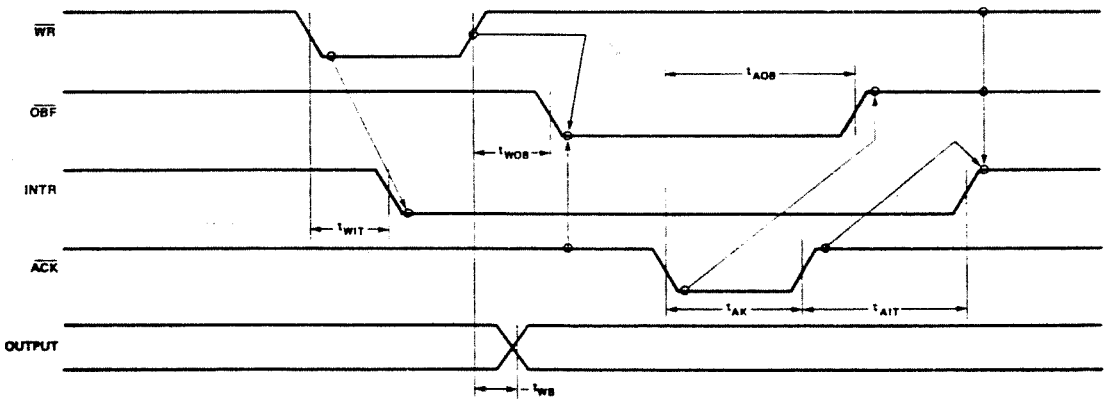


Figure 9. Mode 1 (Strobed Output)

Input Control Signal Definition

STB (Strobe Input). A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into the input latch; in essence, an acknowledgement. IBF is set by STB input being low and is reset by the rising edge of the RD input.

INTR (Interrupt Request)

A "high" on this output can be used to interrupt the CPU when an input device is requesting service. INTR is set by the $\overline{STB}$ is a "one", IBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{RD}$. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

INTE A

Controlled by bit set/reset of PC_4.

INTE B

Controlled by bit set/reset of PC_2.

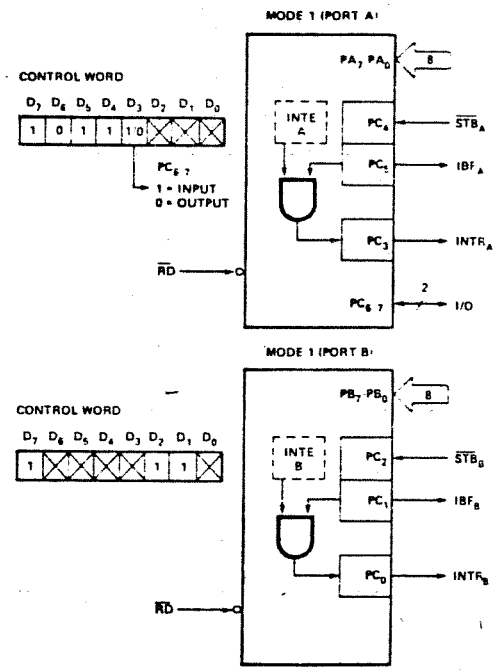


Figure 6. MODE 1 Input

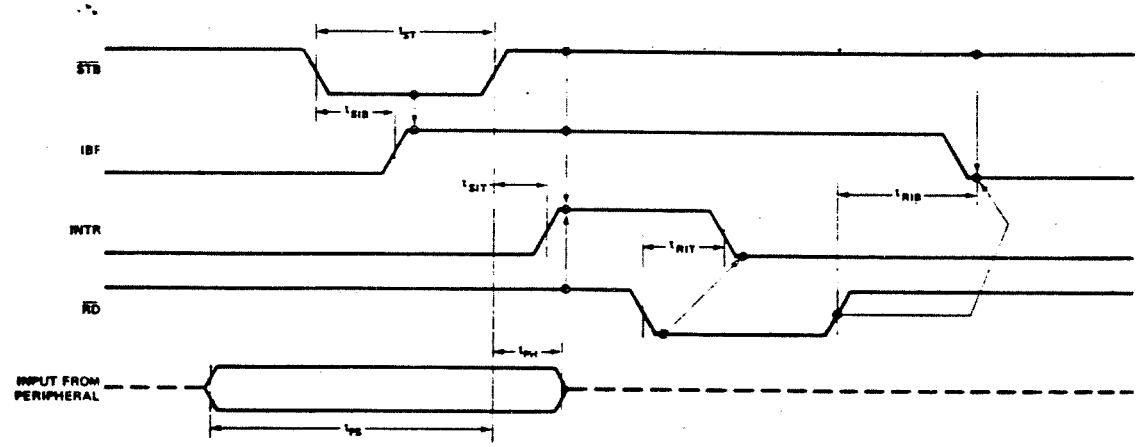


Figure 7. MODE 1 (Strobed Input)

Combinations of MODE 1

Port A and Port B can be individually defined as input or output in Mode 1 to support a wide variety of strobed I/O applications.

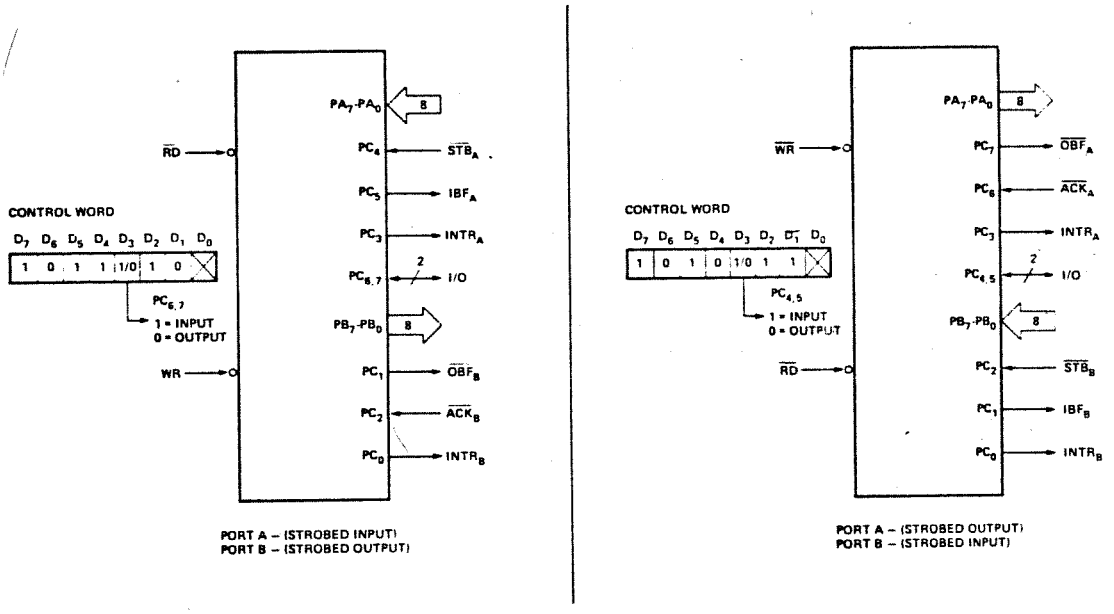


Figure 10. Combinations of MODE 1

Operating Modes

MODE 2 (Strobed Bidirectional Bus I/O). This functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bidirectional bus I/O). "Handshaking" signals are provided to maintain proper bus flow discipline in a similar manner to MODE 1. Interrupt generation and enable/disable functions are also available.

MODE 2 Basic Functional Definitions:

- Used in Group A only.
- One 8-bit, bi-directional bus Port (Port A) and a 5-bit control Port (Port C).
- Both inputs and outputs are latched.
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bi-directional bus port (Port A).

Bidirectional Bus I/O Control Signal Definition

INTR (Interrupt Request). A high on this output can be used to interrupt the CPU for both input or output operations.

Output Operations

$\overline{OBF}$ (Output Buffer Ful). The OBF output will go "low" to indicate that the CPU has written data out to port A.

$\overline{ACK}$ (Acknowledge). A "low" on this input enables the tri-state output buffer of port A to send out the data. Otherwise, the output buffer will be in the high impedance state.

INTE 1 (The INTE Flip-Flop Associated with OBF). Controlled by bit set/reset of PC_6.

Input Operations

$\overline{STB}$ (Strobe Input)

STB (Strobe Input). A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F). A "high" on this output indicates that data has been loaded into the input latch.

INTE 2 (The INTE Flip-Flop Associated with IBF). Controlled by bit set/reset of PC_4.

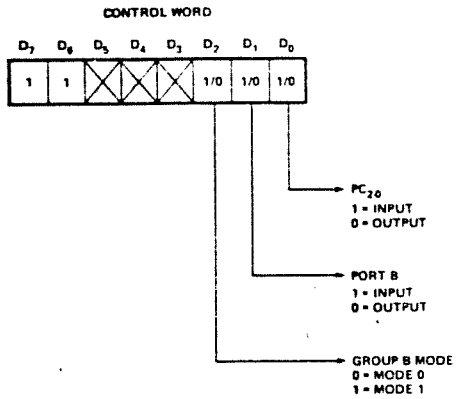


Figure 11. MODE Control Word

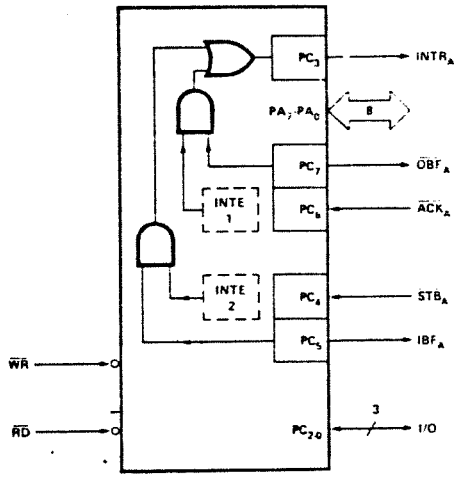


Figure 12. MODE 2

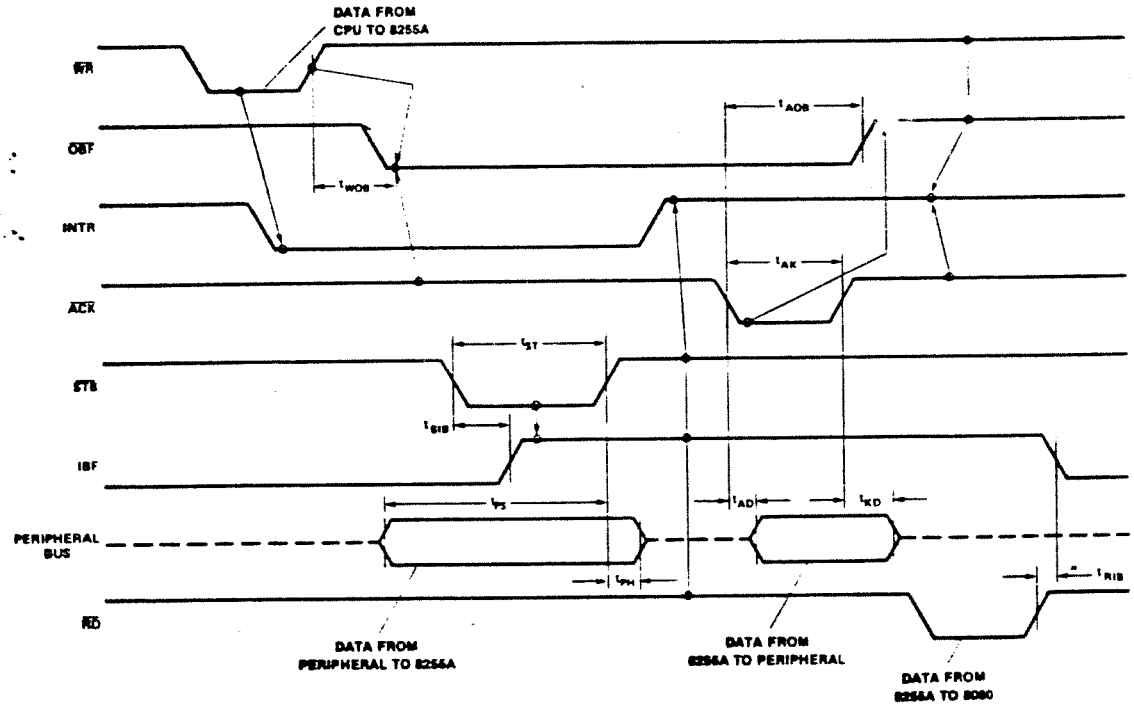


Figure 13. MODE 2 (Bidirectional)

NOTE: Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ and $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
(INTR = IBF • $\overline{MASK}$ • $\overline{STB}$ • $\overline{RD}$ + $\overline{OBF}$ • $\overline{MASK}$ • $\overline{ACK}$ • $\overline{WR}$)

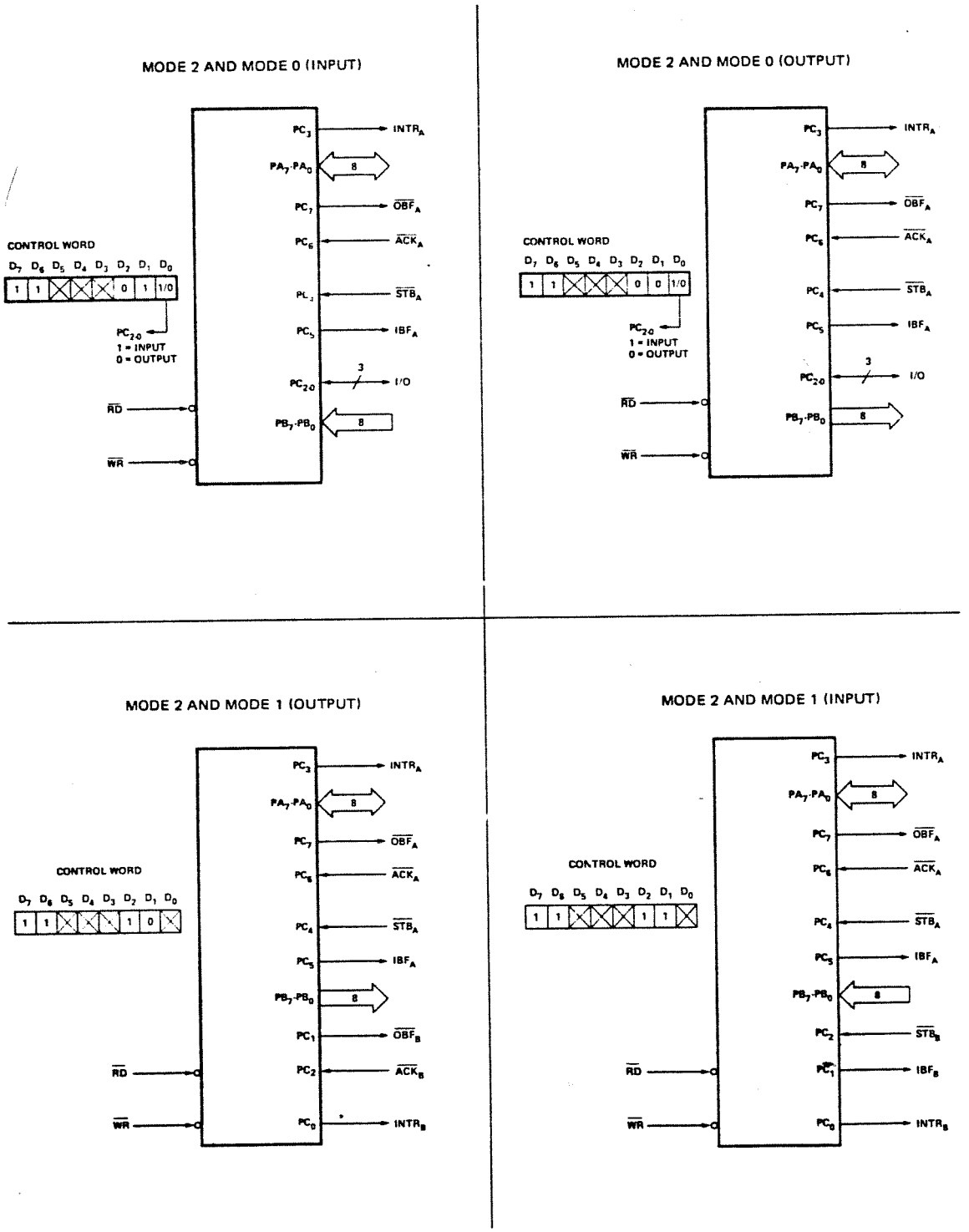


Figure 14. MODE 2 Combinations

Mode Definition Summary

	MODE 0		MODE 1		MODE 2
	IN	OUT	IN	OUT	GROUP A ONLY
PA_0	IN	OUT	IN	OUT	↔
PA_1	IN	OUT	IN	OUT	↔
PA_2	IN	OUT	IN	OUT	↔
PA_3	IN	OUT	IN	OUT	↔
PA_4	IN	OUT	IN	OUT	↔
PA_5	IN	OUT	IN	OUT	↔
PA_6	IN	OUT	IN	OUT	↔
PA_7	IN	OUT	IN	OUT	↔
PB_0	IN	OUT	IN	OUT	—
PB_1	IN	OUT	IN	OUT	—
PB_2	IN	OUT	IN	OUT	—
PB_3	IN	OUT	IN	OUT	—
PB_4	IN	OUT	IN	OUT	—
PB_5	IN	OUT	IN	OUT	—
PB_6	IN	OUT	IN	OUT	—
PB_7	IN	OUT	IN	OUT	—
PC_0	IN	OUT	$INTR_B$	$INTR_B$	I/O
PC_1	IN	OUT	IBF_B	$\overline{OBF}_B$	I/O
PC_2	IN	OUT	$\overline{STB}_B$	$\overline{ACK}_B$	I/O
PC_3	IN	OUT	$INTR_A$	$INTR_A$	$INTR_A$
PC_4	IN	OUT	$\overline{STB}_A$	I/O	$\overline{STB}_A$
PC_5	IN	OUT	IBF_A	I/O	IBF_A
PC_6	IN	OUT	I/O	$\overline{ACK}_A$	$\overline{ACK}_A$
PC_7	IN	OUT	I/O	$\overline{OBF}_A$	$\overline{OBF}_A$

MODE 0 OR MODE 1 ONLY

Special Mode Combination Considerations

There are several combinations of modes when not all of the bits in Port C are used for control or status. The remaining bits can be used as follows:

- If Programmed as Inputs –
  All input lines can be accessed during a normal Port C read.

  If Programmed as Outputs –
  Bits in C upper (PC_7-PC_4) must be individually accessed using the bit set/reset function.

  Bits in C lower (PC_3-PC_0) can be accessed using the bit set/reset function or accessed as a threesome by writing into Port C.

Source Current Capability on Port B and Port C

Any set of eight output buffers, selected randomly from Ports B and C can source 1mA at 1.5 volts. This feature allows the 8255 to directly drive Darlington type drivers and high-voltage displays that require such source current.

Reading Port C Status

In Mode 0, Port C transfers data to or from the peripheral device. When the 8255 is programmed to function in Modes 1 or 2, Port C generates or accepts "hand-shaking" signals with the peripheral device. Reading the contents of Port C allows the programmer to test or verify the "status" of each peripheral device and change the program flow accordingly.

There is no special instruction to read the status information from Port C. A normal read operation of Port C is executed to perform this function.

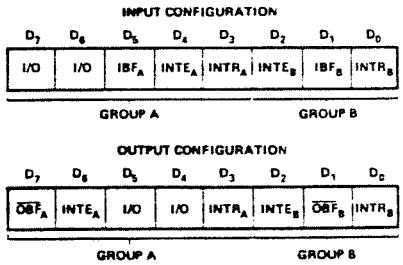


Figure 15. MODE 1 Status Word Format

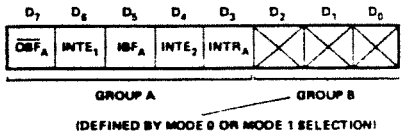


Figure 16. MODE 2 Status Word Format

APPLICATIONS OF THE 8255A

The 8255A is a very powerful tool for interfacing peripheral equipment to the microcomputer system. It represents the optimum use of available pins and is flexible enough to interface almost any I/O device without the need for additional external logic.

Each peripheral device in a microcomputer system usually has a "service routine" associated with it. The routine manages the software interface between the device and the CPU. The functional definition of the 8255A is programmed by the I/O service routine and becomes an extension of the system software. By examining the I/O devices interface characteristics for both data transfer and timing, and matching this information to the examples and tables in the detailed operational description, a control word can easily be developed to initialize the 8255A to exactly "fit" the application. Figures 17 through 23 present a few examples of typical applications of the 8255A.

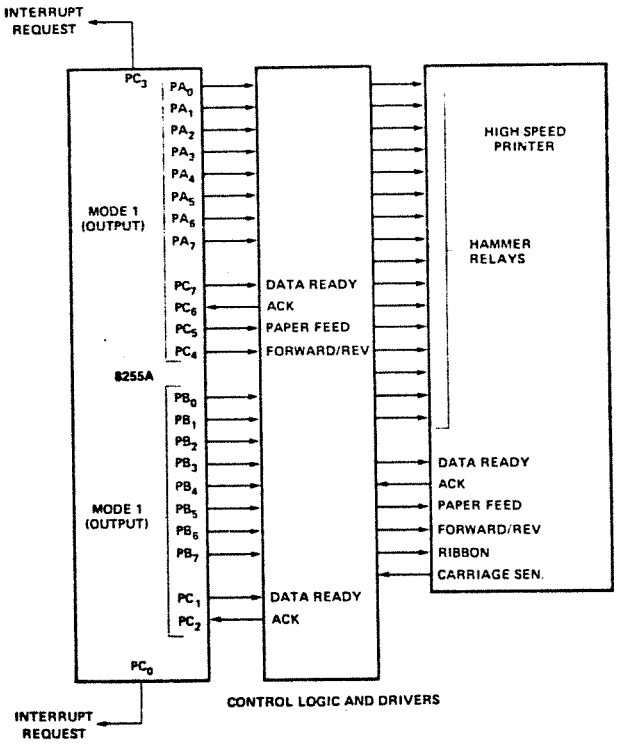


Figure 17. Printer Interface

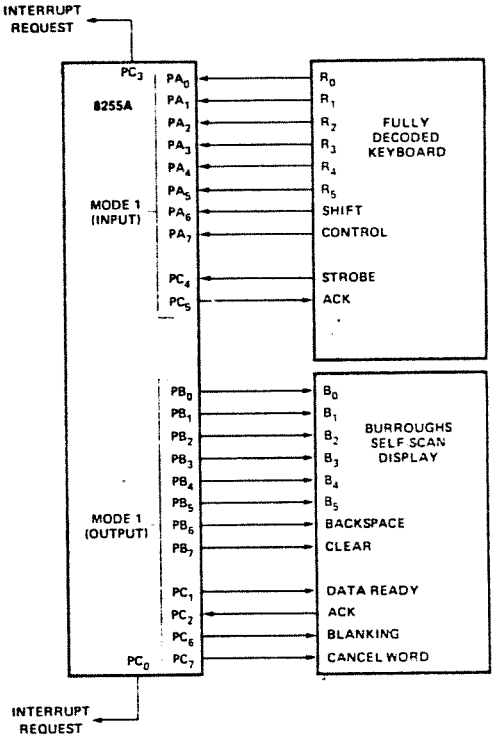


Figure 18. Keyboard and Display Interface

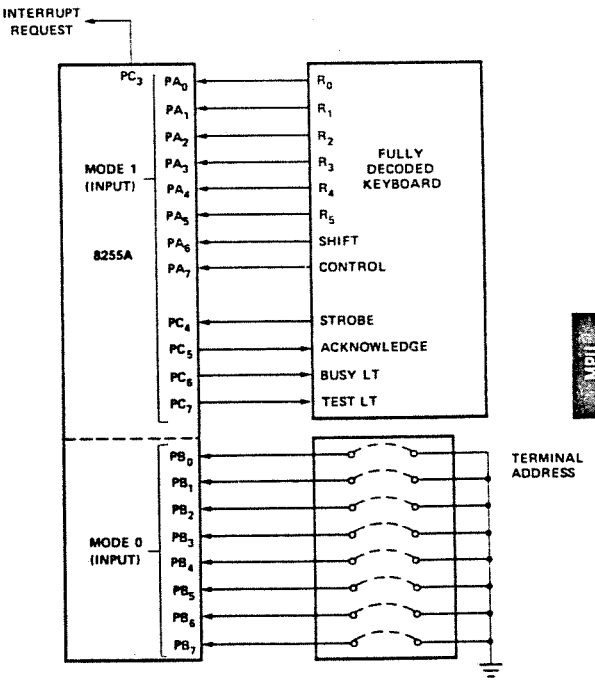


Figure 19. Keyboard and Terminal Address Interface

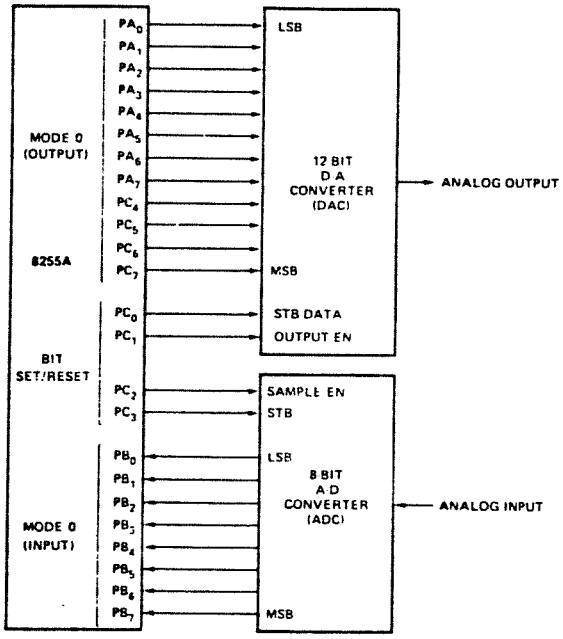


Figure 20. Digital to Analog, Analog to Digital

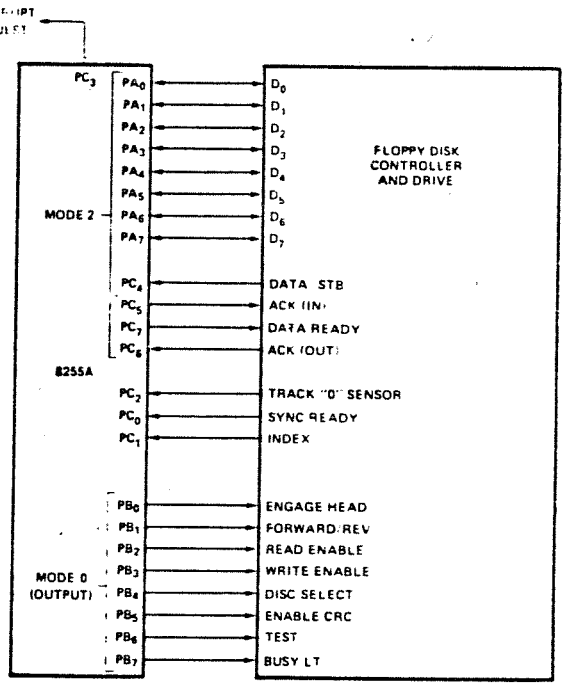


Figure 22. Basic Floppy Disc Interface

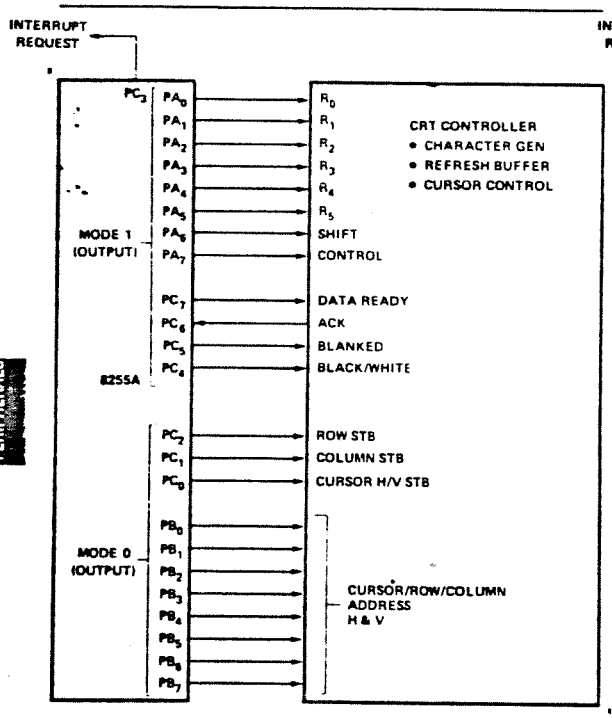


Figure 21. Basic CRT Controller Interface

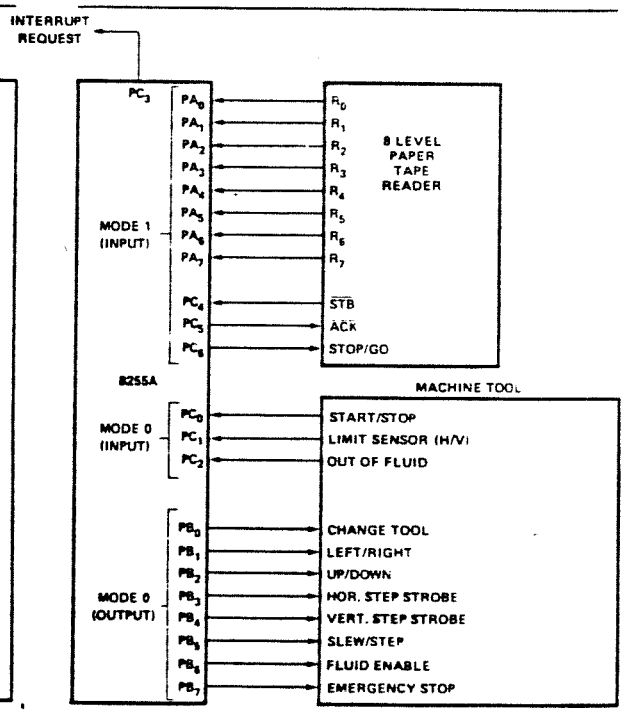


Figure 23. Machine Tool Controller Interface

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias. . . . . 0°C to 70°C
Storage Temperature . . . . . .–65°C to +150°C
Voltage on Any Pin
With Respect to Ground . . . . . –0.5V to +7V
Power Dissipation . . . . . 1 Watt

**COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

D.C. CHARACTERISTICS

T_A = 0°C to 70°C, V_{CC} = +5V ±5%; GND = 0V

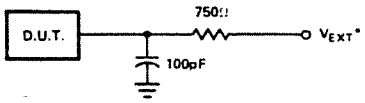
SYMBOL	PARAMETER	MIN.	MAX.	UNIT	TEST CONDITIONS
V_{IL}	Input Low Voltage	-0.5	0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC}	V	
V_{OL} (DB)	Output Low Voltage (Data Bus)		0.45	V	I_{OL} = 2.5mA
V_{OL}(PER)	Output Low Voltage (Peripheral Port)		0.45	V	I_{OL} = 1.7mA
V_{OH}(DB)	Output High Voltage (Data Bus)	2.4		V	I_{OH} = -400μA
V_{OH}(PER)	Output High Voltage (Peripheral Port)	2.4		V	I_{OH} = -200μA
I_{DAR}[1]	Darlington Drive Current	-1.0	-4.0	mA	R_{EXT} = 750Ω; V_{EXT} = 1.5V
I_{CC}	Power Supply Current		120	mA	
I_{IL}	Input Load Current		±10	μA	V_{IN} = V_{CC} to 0V
I_{OFL}	Output Float Leakage		±10	μA	V_{OUT} = V_{CC} to 0V

Note 1: Available on any 8 pins from Port B and C.

CAPACITANCE

T_A = 25°C; V_{CC} = GND = 0V

SYMBOL	PARAMETER	MIN.	TYP.	MAX.	UNIT	TEST CONDITIONS
C_{IN}	Input Capacitance			10	pF	fc = 1MHz
$C_{I/O}$	I/O Capacitance			20	pF	Unmeasured pins returned to GND



*V_{EXT} is set at various voltages during testing to guarantee the specification.

Figure 24. Test Load Circuit (for dB)

A.C. CHARACTERISTICS

T_A = 0°C to 70°C; V_{CC} = +5V ±5%; GND = 0V

NOTE:
The 8255A-5 specifications are not final. Some parametric limits are subject to change

Bus Parameters

Read:

SYMBOL	PARAMETER	8255A MIN.	8255A MAX.	8255A-5 MIN.	8255A-5 MAX.	UNIT
t_{AR}	Address Stable Before READ	0		0		ns
t_{RA}	Address Stable After READ	0		0		ns
t_{RR}	READ Pulse Width	300		300		ns
t_{RD}	Data Valid From READ[1]		250		200	ns
t_{DF}	Data Float After READ	10	150	10	100	ns
t_{RV}	Time Between READs and/or WRITEs	850		850		ns

Write:

SYMBOL	PARAMETER	8255A MIN.	8255A MAX.	8255A-5 MIN.	8255A-5 MAX.	UNIT
t_{AW}	Address Stable Before WRITE	0		0		ns
t_{WA}	Address Stable After WRITE	20		20		ns
t_{WW}	WRITE Pulse Width	400		300		ns
t_{DW}	Data Valid to WRITE (T.E.)	100		100		ns
t_{WD}	Data Valid After WRITE	30		30		ns

Other Timings:

SYMBOL	PARAMETER	8255A MIN.	8255A MAX.	8255A-5 MIN.	8255A-5 MAX.	UNIT
t_{WB}	WR = 1 to Output[1]		350		350	ns
t_{IR}	Peripheral Data Before RD	0		0		ns
t_{HR}	Peripheral Data After RD	0		0		ns
t_{AK}	ACK Pulse Width	300		300		ns
t_{ST}	STB Pulse Width	500		500		ns
t_{PS}	Per. Data Before T.E. of STB	0		0		ns
t_{PH}	Per. Data After T.E. of STB	180		180		ns
t_{AD}	ACK = 0 to Output[1]		300		300	ns
t_{KD}	ACK = 1 to Output Float	20	250	20	250	ns
t_{WOB}	WR = 1 to OBF = 0[1]		650		650	ns
t_{AOB}	ACK = 0 to OBF = 1[1]		350		350	ns
t_{SIB}	STB = 0 to IBF = 1[1]		300		300	ns
t_{RIB}	RD = 1 to IBF = 0[1]		300		300	ns
t_{RIT}	RD = 0 to INTR = 0[1]		400		400	ns
t_{SIT}	STB = 1 to INTR = 1[1]		300		300	ns
t_{AIT}	ACK = 1 to INTR = 1[1]		350		350	ns
t_{WIT}	WR = 0 to INTR = 0[1]		850		850	ns

Notes: 1. Test Conditions: 8255A: C_L = 100pF; 8255A-5: C_L = 150pF.
2. Period of Reset pulse must be at least 50μs during or after power on. Subsequent Reset pulse can be 500 ns min.

MPU PERIPHERALS

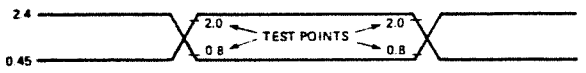


Figure 25. Input Waveforms for A.C. Tests

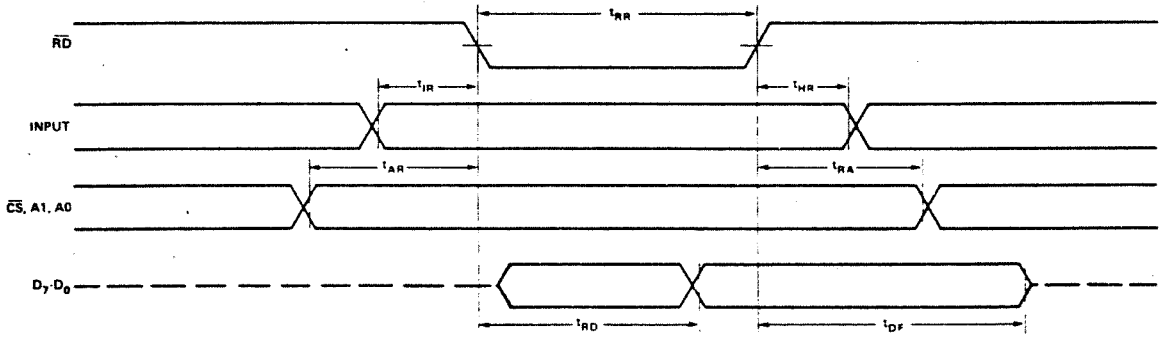


Figure 26. MODE 0 (Basic Input)

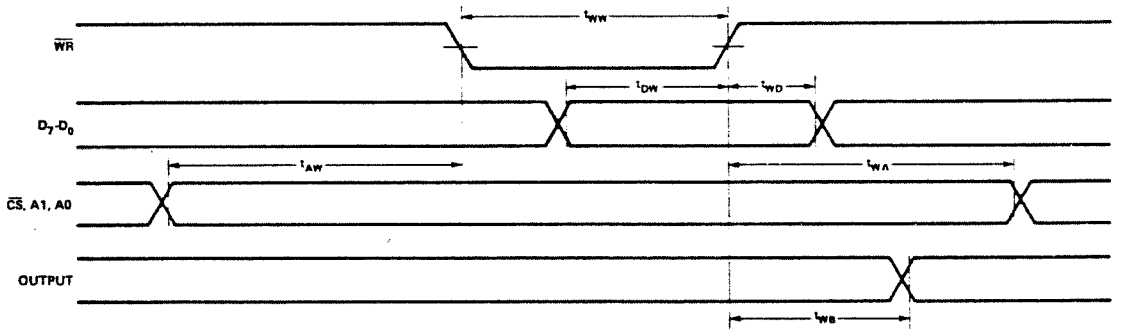


Figure 27. MODE 0 (Basic Output)

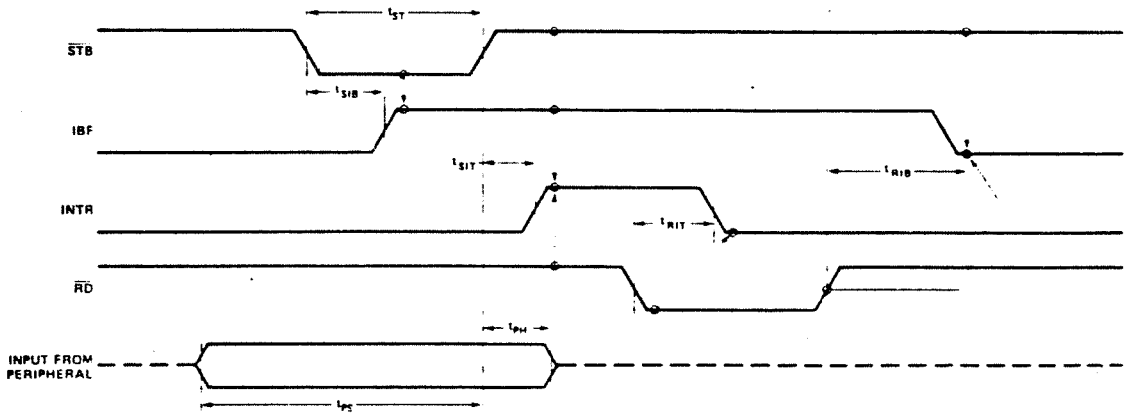


Figure 28. MODE 1 (Strobed Inut)

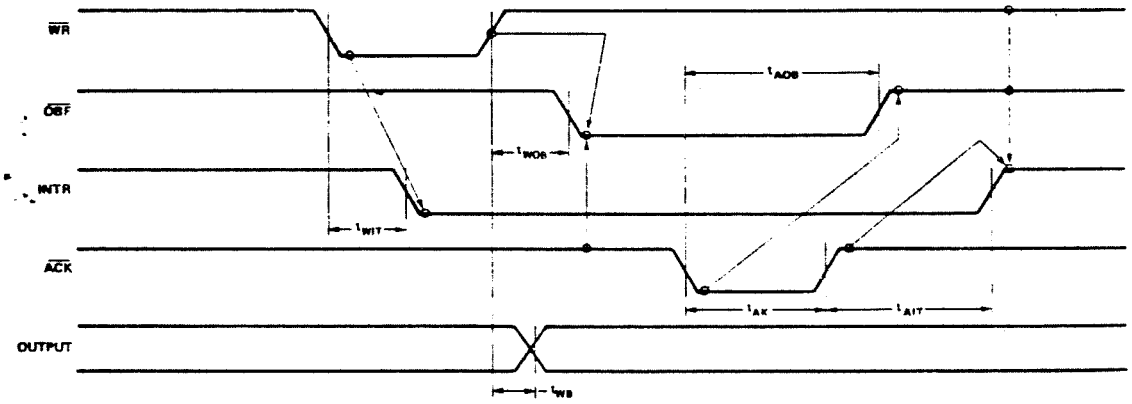


Figure 29. MODE 1 (Strobed Output)

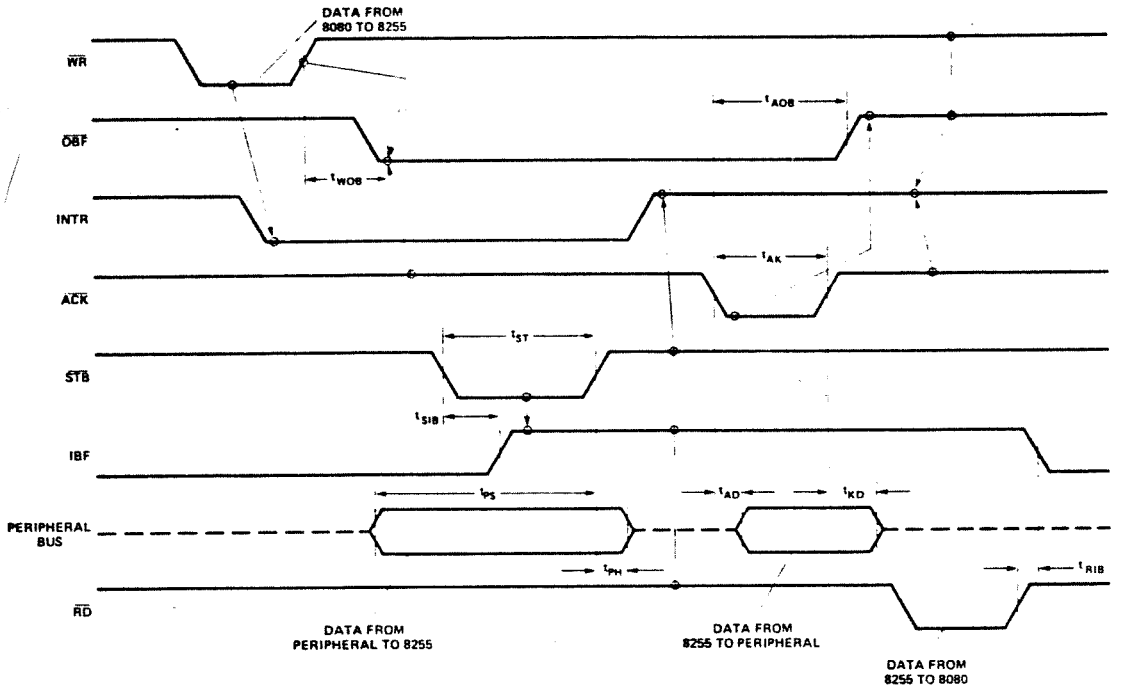


Figure 30. MODE 2 (Bidirectional)

NOTE: Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ and $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
(INTR = IBF • $\overline{MASK}$ • $\overline{STB}$ • $\overline{RD}$ + $\overline{OBF}$ • $\overline{MASK}$ • $\overline{ACK}$ • $\overline{WR}$)

APPENDIX 8

M O N I T O R P R O G R A M

- S O U R C E L I S T I N G -

Reproduced Courtesy of

MULTITECH INDUSTRIAL CORPORATION
977 MIN SHEN E. ROAD
TAIPEI 105 TAIWAN
Republic of China

E & L modifications at addresses

0037 Hex
&
077B Hex thru 07A4 Hex

***See page 2 of source listing

***THIS CODE REPLACES THE CODE SHOWN IN SOURCE LISTING

```
LOC   OBJ CODE M STMT SOURCE STATEMENT

0037  71         191          DEFB   ZSUM

                 2490  KEYTAB:
077B  03         2491  K0     DEFB   12H     ;GO
077C  07         2492  K1     DEFB   1EH     ;DUMP
077D  0B         2493  K2     DEFB   1CH     ;COPY
077E  0F         2494  K3     DEFB   10H     ;NEXT
077F  20         2495  K4     DEFB   20H     ;NOT USED
0780  21         2496  K5     DEFB   21H     ;NOT USED
0781  02         2497  K6     DEFB   13H     ;STEP
0782  06         2498  K7     DEFB   1FH     ;LOAD
0783  0A         2499  K8     DEFB   17H     ;DELETE
0784  0E         2500  K9     DEFB   11H     ;PREV
0785  22         2501  K0A    DEFB   22H     ;NOT USED
0786  23         2502  K0B    DEFB   23H     ;NOT USED
0787  01         2503  K0C    DEFB   18H     ;PC
0788  05         2504  K0D    DEFB   1AH     ;CLR BRKPT
0789  09         2505  K0E    DEFB   16H     ;INSERT
078A  0D         2506  K0F    DEFB   14H     ;DATA
078B  13         2507  K10    DEFB   09H     ;HEX 9
078C  1F         2508  K11    DEFB   0CH     ;HEX C
078D  00         2509  K12    DEFB   1BH     ;REG
078E  04         2510  K13    DEFB   15H     ;SBR
078F  08         2511  K14    DEFB   1DH     ;RE. A
0790  0C         2512  K15    DEFB   19H     ;ADDR
0791  12         2513  K16    DEFB   0DH     ;HEX D
0792  1E         2514  K17    DEFB   08H     ;HEX 8
0793  1A         2515  K18    DEFB   06H     ;HEX 6
0794  18         2516  K19    DEFB   03H     ;HEX 3
0795  1B         2517  K1A    DEFB   07H     ;HEX 7
0796  19         2518  K1B    DEFB   0FH     ;HEX F
0797  17         2519  K1C    DEFB   05H     ;HEX 5
0798  1D         2520  K1D    DEFB   04H     ;HEX 4
0799  15         2521  K1E    DEFB   02H     ;HEX 2
079A  11         2522  K1F    DEFB   0AH     ;HEX A
079B  14         2523  K20    DEFB   0BH     ;HEX B
079C  10         2524  K21    DEFB   0EH     ;HEX E
079D  16         2525  K22    DEFB   01H     ;HEX 1
079E  1C         2526  K23    DEFB   00H     ;HEX 0
                 2527  ;
                 2528  ; The new version has 2 decimal points in the
                 2529  ; initial display pattern.
                 2530  ;
079F  30         2531  MPF_I  DEFB   00H     ;BLANK
07A0  02         2532         DEFB   B6H     ;Y
07A1  02         2533         DEFB   B3H     ;d
07A2  0F         2534         DEFB   3FH     ;A
07A3  1F         2535         DEFB   8FH     ;E
07A4  A1         2536         DEFB   03H     ;r
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT

 1  ;*********************************************************
 2  ;*                                                       *
 3  ;*    COPYRIGHT , MULTITECH INDUSTRIAL CORP. 1981        *
 4  ;*    All right reserved.                                *
 5  ;*    No part of this software may be copied without     *
 6  ;*    the express written consent of MULTITECH           *
 7  ;*    INDUSTRIAL CORP.                                   *
 8  ;*                                                       *
 9  ;*********************************************************
10  ;
11  ;
12  ;
13  ;
14  ;
15  P8255   EQU     03H     ;8255 I control port
16  DIGIT   EQU     02H     ;8255 I port C
17  SEG7    EQU     01H     ;8255 I port B
18  KIN     EQU     00H     ;8255 I port A
19  PWCODE  EQU     0A5H    ;Power-up code
20  ZSUM    EQU     71H     ;This will make the sum of all
21                          ;monitor codes to be zero.
22
23  ; The following EQUATEs are used for timing.  Their values
24  ; depend on the CPU clock frequency. (In this version, the
25  ; crystal frequency is 1.79 MHz.)
26
27  COLDEL  EQU     201     ;Column delay time for routine
28                          ;SCAN and SCAN1.
29  F1KHZ   EQU     65      ;Delay count for 1K Hz square wave,
30                          ;used by routin TONE1K.
31  F2KHZ   EQU     31      ;Delay count for 2K Hz square wave,
32                          ;used by routine TONE2K.
33  MPERIOD EQU     42      ;1K Hz and 2K Hz threshold, used by
34                          ;tape input routine PERIOD.
35
36  ; The following EQUATEs are for tape modulation.
37  ; If the quality of tape recorder is good, the user may
38  ; change '4 4 2 8' to '2 2 1 4'.  This will double
39  ; the tape data rate.
40  ; If the quality of tape recorder is poor, the user may
41  ; change '4 4 2 8 ' to '6 6 3 12'.  This will improve
42  ; error performance but slow down the data rate.
43  ; Although the data format is changed, the tape is still
44  ; compatible in each case, because only the ratio is
45  ; detected in the Tape-read.
46
47  ONE_1K  EQU     4
48  ONE_2K  EQU     4
49  ZERO_1K EQU     2
50  ZERO_2K EQU     8
51
52  ;*********************************************************
53  ; I/O port assignment: (8255 I)
54
55  ; port A (address 00H):
56  ;       bit 7 -- tape input
57  ;       bit 6 -- 'USER KEY' on keyboard, active low
58  ;       bit 5-0 row of keyboard matrix input ,active low
59  ; port B (address 01H): 7 segaments of LED, active high
60  ;       bit 7 -- segament d
61  ;       bit 6 -- decimal point
62  ;       bit 5 -- segament c
63  ;       bit 4 -- segament b
64  ;       bit 3 -- segament a
65  ;       bit 2 -- segament f
66  ;       bit 1 -- segament g
67  ;       bit 0 -- segament e
68  ; port C (address 02H):
69  ;       bit 7 -- tape & tone output
70  ;       bit 6 -- BREAK enable. NMI (CPU pin 17) will goes to
71  ;                low 5 M1's (machine cycle one) after this
72  ;                bit goes to low.  (This bit is connected to
73  ;                the reset input of external counter.)
74  ;       bit 5¢0 -- columns of keyboard and display matrix,
75  ;               active high.  Bit 5 is the leftmost column.
76
77  ;*********************************************************
78  ;  -- reset --
79  ; There are two cases that will generate a RESET signal:
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 80  ;     (i) power-up
                 81  ;    (ii) 'RS' key pressed
                 82  ; In both cases, the follow actions will be taken:
                 83  ;    a) disable interrupt, set interrupt mode to 0
                 84  ;       set I register to 00 and start execution
                 85  ;       at address 0000 (by Z80 CPU itself).
                 86  ;    b) initial user's PC to the lowest RAM address;
                 87  ;    c) set user's SP to 1F9FH;
                 88  ;    d) set user's I register to 00 and disable user's
                 89  ;       interrupt flip-flop;
                 90  ; In addition, subroutine INI will be called on power-up
                 91  ; reset, which has the following effects:
                 92  ;    e) disable BREAK POINT;
                 93  ;    f) set the contents of location 1FEEH  1FEFH to 66 and
                 94  ;       and 00 respectively. This will make instruction RST
                 95  ;       38H (opcode FF) have the same effect as BREAK.
                 96  ; Memory location POWERUP is used to distinguish power-up
                 97  ; from RS-key.  (POWERUP) contains a random data when
                 98  ; power-up and contains PWCODE (0A5H) thereafter.
                 99
0000   0600     100         LD      B,0
0002   10FE     101         DJNZ    $        ;Power-up delay
                102
                103  ; Initial 8255 to mode 0 with port A input, port B and C
                104  ; output. The control word is 90H.
                105
0004   3E90     106         LD      A,10010000B
0006   D303     107         OUT     (P8255),A
                108
                109  ; When the control word is sent out to 8255, all output
                110  ; ports are cleared to 0.  It is necessary to disable
                111  ; BREAK and deactivate all I/O by sending 0C0H to
                112  ; port C.
                113
0008   3EC0     114         LD      A,0C0H
000A   D302     115         OUT     (DIGIT),A
000C   31AF1F   116         LD      SP,SYSSTK        ;initial system stack
                117
                118  ; If the content of location POWERUP is not equal to
                119  ; PWCODE, call subroutine INI. Continue otherwise.
                120
000F   3A51F    121         LD      A,(POWERUP)
0012   FEA5     122         CP      PWCODE
0014   C4C103   123         CALL    NZ,INI
                124
                125  ; Determine the lowest RAM address by checking whether
                126  ; address 1000H is RAM.  If yes, set user's PC to this
                127  ; value.  Otherwise, set it to 1800H.
                128
0017   210010   129         LD      HL,1000H
001A   CDF605   130         CALL    RAMCHK
001D   2802     131         JR      Z,PREPC
001F   2618     132         LD      H,18H
0021   22DC1F   133  PREPC  LD      (USERPC),HL
0024   2600     134         LD      H,0
                135
                136  ; Address 28H and 30H are reserved for BREAK (RST 28H)
                137  ; and software BREAK (RST 30H).  Skip these area, monitor
                138  ; program resumes at RESET1.
                139
0026   180A     140         JR      RESET1
                141  ;
                142  ;*****************************************************
0028            143  RST28   ORG     28H
                144  ; Address 28H is the entry point of BREAK trap.
                145  ; If a location is set as a BREAK point, the monitor
                146  ; will change the content of this location to C7 (RST 28H)
                147  ; before transfering control to user's program.
                148  ; In execution of user's program, a trap will occur if
                149  ; user's PC passes this location.  The monitor then takes
                150  ; over control and the content of BREAK address
                151  ; will be restored.  Monitor takes care of everything
                152  ; and makes the whole mechanism transparant to the user.
                153  ; The return address pushed onto stack is the PC after
                154  ; executing RST 28H.  The original break address should
                155  ; be one less than that.  The following 3 instructions
                156  ; decrease the content of (SP) by one without changing
                157  ; HL.
                158
0028   E3       159         EX      (SP),HL
0029   2B       160         DEC     HL
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
002A  E3          161          EX      (SP),HL
002B  22E81F      162          LD      (HLTEMP),HL
002E  180E        163          JR      CONT28
                  164  ;
                  165  ;*************************************************************
0030              166  RST30   ORG     30H
                  167
                  168  ; Instruction RST 30H (opcode F7) is usually used as:
                  169  ;   i) Software break;
                  170  ;  ii) Terminator of user's program.
                  171  ; The effect of this instruction is to save all user's
                  172  ; registers and return to monitor.
                  173
0030  1834        174          JR      NMI
                  175
                  176  ;*************************************************************
                  177  ; This is a part of reset routine.  Address 0028 and
                  178  ; 0030 are reserved for break point.  Reset routine
                  179  ; skips this area and resumes here.
                  180  ;
0032  22D21F      181  RESET1  LD      (USERIF),HL      ;set user's I register and
                  182                                   ;interrupt flip flop to 0
0035  181D        183          JR      RESET2  ;monitor resumes at RESET2
                  184
                  185  ;*************************************************************
                  186
                  187  ; The following byte makes the sum of the monitor
                  188  ; code in ROM zero. ROMTEST is a self-checking routine.
                  189  ; This routine requires the sum of ROM to be zero.
                  190
0037  71          191          DEFB    ZSUM
                  192  ;
                  193  ;*************************************************************
0038              194  RST38   ORG     38H
                  195
                  196  ; Entry point of RST 38H (opcode FF) or mode 1 interrupt.
                  197  ; Fetch the address stored in location 1FEE and 1FEF,
                  198  ; then jump to this address.  Initially, 1FEE and 1FEF
                  199  ; are set to 0066.  So RST 38 will have the same effect
                  200  ; as software break.  By changing the content of 1FEE
                  201  ; and 1FEF, the user can define his or her own service
                  202  ; routine.
                  203  ; The next three instructions push the contents of 1FEE
                  204  ; and 1FEF to stack without changing any registers.
                  205
0038  E5          206          PUSH    HL
0039  2AEE1F      207          LD      HL,(IM1AD)
003C  E3          208          EX      (SP),HL
                  209
                  210  ; The top of the stack is now the address of user
                  211  ; defined service routine.  Pop out this address then
                  212  ; branch to it.
                  213
003D  C9          214          RET
                  215  ;
                  216  ;*************************************************************
                  217  CONT28:
                  218  ; This is a part of break service routine.  It continues
                  219  ; the program at RST28.
                  220
003E  32E71F      221          LD      (ATEMP),A
                  222
                  223  ; The monitor has changed the content of user's
                  224  ; program at break address.  The next 3 instructions
                  225  ; restored the destroyed content.  BRAD contains the
                  226  ; break address, BRDA contains the original data at
                  227  ; break address.
                  228
0041  2AE01F      229          LD      HL,(BRAD)
0044  3AE21F      230          LD      A,(BRDA)
0047  77          231          LD      (HL),A
                  232
                  233  ; Send break enable signal to hardware counter.
                  234  ; A nonmaskable interrupt will be issued at the 5th M1's.
                  235
0048  3E80        236          LD      A,10000000B
004A  D302        237          OUT     (DIGIT),A
004C  3AE71F      238          LD      A,(ATEMP)        ; 1st M1
004F  2AE81F      239          LD      HL,(HLTEMP)      ; 2nd M1
0052  00          240          NOP                      ; 3rd M1
0053  C9          241          RET                      ; 4th M1
```

```
LOC    OBJ CODE M STMT SOURCE STATEMENT
                  242
                  243  ; Return to user's program.  Execute the instruction
                  244  ; at break address.  After finishing one instruction,
                  245  ; a nonmaskable interrupt happens and control is
                  246  ; transferred to the monitor again.
                  247  ;
                  248  RESET2:
0054   219F1F     249          LD      HL,USERSTK
0057   22D01F     250          LD      (USERSP),HL      ;set user's SP
005A   AF         251          XOR     A
005B   32E61F     252          LD      (TEST),A
                  253
                  254  ; TEST is a flag for monitor's own use.  Illegal key-in
                  255  ; blanking (bit 7 of TEST) and automatic leading zero
                  256  ; (bit 0) use this flag.  Clear it here.
                  257
005E   DD219F07   258          LD      IX,MPF_I         ;Initial display pattern.
                  259
                  260  ; Address 0066 is the address for nonmaskable interrupt.
                  261  ; Skip this area, monitor resumes at SETST0
                  262
0062   C3D000     263          JP      SETST0
                  264  ;
                  265  ;*****************************************************
0066              266  NMI     ORG     66H
                  267
                  268  ; Entry point of nonmaskable interrupt.  NMI will occur
                  269  ; when MONI key is pressed or when user's program is
                  270  ; breaked.  The service routine which starts here saves all
                  271  ; user's registers and status.  It also check the validity
                  272  ; of user's SP.
                  273
0066   32E71F     274          LD      (ATEMP),A        ;save A register
0069   3E90       275          LD      A,10010000B
006B   D303       276          OUT     (P8255),A        ;set 8255 to mode 0.
                  277                                   ;Port A input; B,C output.
006D   3EC0       278          LD      A,0C0H
006F   D302       279          OUT     (DIGIT),A        ;disable break and LED's
0071   3AE71F     280          LD      A,(ATEMP)        ;restore A register
0074   22E81F     281  RGSAVE  LD      (HLTEMP),HL      ;save register HL
0077   E1         282          POP     HL       ;get return address from stack
0078   22DE1F     283          LD      (ADSAVE),HL      ;Save return address into
                  284                                   ;ADSAVE.
007B   22DC1F     285          LD      (USERPC),HL      ;Set user's PC to return
                  286                                   ;address.
007E   2AE81F     287          LD      HL,(HLTEMP)      ;restore HL register
0081   ED73D01F   288          LD      (USERSP),SP      ;set user's SP to current SP
0085   31D01F     289          LD      SP,USERIY+2      ;save other registers by
0088   FDE5       290          PUSH    IY               ;continously pushing them
008A   DDE5       291          PUSH    IX               ;onto stack
008C   D9         292          EXX
008D   E5         293          PUSH    HL
008E   D5         294          PUSH    DE
008F   C5         295          PUSH    BC
0090   D9         296          EXX
0091   08         297          EX      AF,AF'
0092   F5         298          PUSH    AF
0093   08         299          EX      AF,AF'
0094   E5         300          PUSH    HL
0095   D5         301          PUSH    DE
0096   C5         302          PUSH    BC
0097   F5         303          PUSH    AF
                  304
                  305  ; The next two instructions save I register.
                  306  ; The interrupt flip-flop (IFF2) is copied into
                  307  ; parity flag (P/V) by instruction LD  A,I.
                  308  ; The interrupt status (enabled or disabled)
                  309  ; can be determined by testing parity flag.
                  310
0098   ED57       311          LD      A,I
009A   32D31F     312          LD      (USERIF+1),A
                  313
                  314  ; The next four instructions save IFF2 into
                  315  ; user's IFF.
                  316
009D   3E00       317          LD      A,0
009F   E2A400     318          JP      PO,SETIF         ;PO -- P/V = 0
00A2   3E01       319          LD      A,1
00A4   32D21F     320  SETIF   LD      (USERIF),A
                  321  ;
00A7   31AF1F     322          LD      SP,SYSSTK        ;set SP to system stack
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 323
                 324  ; The next 8 instructions check user's SP.
                 325  ; If the user's SP points to a location not
                 326  ; in RAM, display ERR-SP.
                 327
00AA  2AD01F     328          LD      HL,(USERSP)
00AD  DD21B507   329          LD      IX,ERR_SP
00B1  2B         330          DEC     HL
00B2  CDF605     331          CALL    RAMCHK
00B5  2019       332          JR      NZ,SETSTO
00B7  2B         333          DEC     HL
00B8  CDF605     334          CALL    RAMCHK
00BB  2013       335          JR      NZ,SETSTO
                 336
                 337  ; If the user's stack and system stack are
                 338  ; overlayed, display SYS-SP.  This checking
                 339  ; is done by the following  instructions.
                 340
00BD  DD21AF07   341          LD      IX,SYS_SP
00C1  00         342          NOP
00C2  00         343          NOP
                 344
00C3  1162E0     345          LD      DE,-USERSTK+1
00C6  19         346          ADD     HL,DE
00C7  3807       347          JR      C,SETSTO
00C9  DD21B61F   348          LD      IX,DISPBF
00CD  37         349          SCF                ;set carry flag to indicate
                 350                             ;the user's SP is legal.
00CE  1804       351          JR      BRRSTO
                 352  ;
                 353  SETSTO:
                 354  ; STATE is a memory location contains the monitor status.
                 355  ; It will be described in detail later.  STATE 0 stands
                 356  ; for fixed display pattern.  The initial pattern 'uPF--1'
                 357  ; or message 'SYS-SP'... belong to this category.  The next
                 358  ; two instruction set STATE to zero.
                 359
00D0  AF         360          XOR     A          ;set A to 0, also clear Carry flag
00D1  32E41F     361          LD      (STATE),A
00D4  3AE21F     362  BRRSTO  LD      A,(BRDA)   ;restore the data at
                 363                             ;break address
00D7  2AE01F     364          LD      HL,(BRAD)
00DA  77         365          LD      (HL),A
                 366
                 367  ; If the user's SP is legal (carry set),
                 368  ; display user's PC and the content at PC.
                 369  ; Otherwise, display fixed message (ERR-SP
                 370  ; or SYS-SP or uPF--1)
00DB  DC0B04     371          CALL    C,MEMDP2
                 372  ;
                 373  ;
                 374  ;*********************************************************
                 375  ; Scan the display and keyboard. When a key is
                 376  ; detected, take proper action  according to the
                 377  ; key pressed.
                 378
                 379  MAIN:
00DE  31AF1F     380          LD      SP,SYSSTK          ;Initial system stack.
00E1  CDFE05     381          CALL    SCAN       ;Scan display and input keys.
                 382                             ;Routine SCAN will not return until
                 383                             ;any key is pressed.
00E4  CDCB06     384          CALL    BEEP ;After a key is detected, there
                 385                          ;will be accompanied with a beep
                 386                          ;sound.
00E7  18F5       387          JR      MAIN       ;Back to MAIN, get more keys and
                 388                             ;execute them.
                 389  ;
                 390  ;
                 391  ;*********************************************************
                 392  KEYEXEC:
                 393
                 394  ; Input key dispatch routine.
                 395  ; This routine uses the key code returned by subroutine
                 396  ; SCAN, which is one byte stored in A register.  The
                 397  ; range of key code is from 00 to 1FH.
                 398
                 399  ; (i) key code = 00 ¢ 0FH :
                 400  ;     These are hexadecimal keys.  Branch to routine KHEX.
                 401
00E9  FE10       402          CP      10H
00EB  3824       403          JR      C,KHEX
```

```
LOC    OBJ CODE M STMT SOURCE STATEMENT
                  404
                  405  ; If the key entered is not hexadecimal, it must be a
                  406  ; function or subfunction key.  This means the previous
                  407  ; numeric entry has terminated.  Bit 0 of TEST flag
                  408  ; must be set at the beginning of a new numeric entry.
                  409  ; This is done by the next two instructions.  (If bit 0
                  410  ; of TEST is set, the data buffer will be automatically
                  411  ; cleared when a hexadecimal key is entered.)
                  412
00ED   21E61F     413          LD      HL,TEST
00F0   CBC6       414          SET     0,(HL)
                  415
                  416  ; (ii) key code = 10H ¢ 17H :
                  417  ;      (+, -, GO, STEP, DATA, SBR, INS, DEL)
                  418  ;      There is no state corresponding to these keys.
                  419  ;      The response of them depends on the current
                  420  ;      state and minor-state. (E.g., the response of '+'
                  421  ;      key depends on the current function.  It is illegal
                  422  ;      when the display is 'uPF--1', but is legal when the
                  423  ;      display is of 'address-data' form.)  In this
                  424  ;      documentation, they are named 'sub-function key'.
                  425  ;      They are all branched by table KSUBFUN and routin
                  426  ;      BRANCH.
                  427
00F2   D610       428          SUB     10H
00F4   FE08       429          CP      8
00F6   213707     430          LD      HL,KSUBFUN
00F9   DAB003     431          JP      C,BRANCH
                  432
                  433 ;(iii) key code = 18H ¢ 1FH
                  434  ;      (PC, Addr, CBr, Reg, Move, Rela, WRtape, RDtape)
                  435  ;      These keys are named 'function key'.  They are
                  436  ;      acceptable at any time.  When they are hit, the
                  437  ;      monitor will unconditionally enter a new state.
                  438  ;      STMINOR contains the minor-state, which is required
                  439  ;      to dispatch some sub-function keys (e.g. +, - ).
                  440
00FC   DD21B61F   441          LD      IX,DISPBF
0100   D608       442          SUB     8
0102   21E41F     443          LD      HL,STATE
0105   77         444          LD      (HL),A   ;set STATE to key-code minus 18H
                  445                           ;The STATE is update here.  It will
                  446                           ;be modified later by local service
                  447                           ;routines if the function-key is PC,
                  448                           ;Addr or CBr.  For other function-
                  449                           ;keys, STATE will not be modified
                  450                           ;later.
0106   21E31F     451          LD      HL,STMINOR
0109   3600       452          LD      (HL),0   ;set STMINOR to 0
010B   214107     453          LD      HL,KFUN  ;KFUN is the base of branch table
                  454                           ;the offset is stored in A
010E   C3B003     455          JP      BRANCH
                  456
                  457  ;
                  458  ;*****************************************************
                  459  ;STATE:
                  460  ;    0=FIX        ;Display fixed pattern, e.g. 'uPF--1'.
                  461  ;    1=AD         ;The hex key entered is interpreted as
                  462                    ;memory address.
                  463  ;    2=DA         ;The hex key entered is interpreted as
                  464                    ;memory data.
                  465  ;    3=RGFIX      ;Display fixed pattern: 'Reg-  ' and
                  466                    ;expect register name to be entered.
                  467  ;    4=MV         ;Expect parameters for 'Move' function.
                  468  ;    5=RL         ;Expect parameters for 'Rela' function.
                  469  ;    6=WT         ;Expect parameters for 'WRtape' func.
                  470  ;    7=RT         ;Expect parameters for 'RDtape' func.
                  471  ;    8=RGAD       ;Hex-key entered will be interpreted as
                  472                    ;address name for registers.
                  473  ;    9=RGDA       ;Hex-key entered will be interpreted as
                  474                    ;data for registers.
                  475  ;
                  476  ; Subroutine name conventions:
                  477  ;     (i) K???? -- K stands for key, ???? is the key name,
                  478  ;                  e.g. KINS corresponds to key 'INS'.  Each
                  479  ;                  time a key ???? is entered, the routine
                  480  ;                  with name K???? will be executed.  All of
                  481  ;                  them are branched by table KFUN or KSUBFUN.
                  482  ;    (ii) H???? -- H stands for hexadecimal, ???? is the
                  483  ;                  current STATE.  For example, routine
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 484  ;                    HDA will be executed if the entered
                 485  ;                    key is hexadecimal and STATE is DA now.
                 486  ;                    These routines are branched by table
                 487  ;                    HTAB.
                 488  ;   (iii) I???? -- I stands for increment (+ key), ???? is
                 489  ;                    the current STATE.  E.g. IMV will be
                 490  ;                    executed when STATE is MV and '+' key
                 491  ;                    is entered.  These routines are branched
                 492  ;                    by table ITAB
                 493  ;    (iv) D???? -- D stands for decrement (- key), ???? is
                 494  ;                    the current STATE.  These routines are
                 495  ;                    branched using table DTAB.
                 496  ;     (v) G???? -- G stands for 'GO' key, ???? is the current
                 497  ;                    STATE.  These routines are branched using
                 498  ;                    table GTAB.
                 499
                 500  ;*************************************************************
                 501
                 502  ; Hexadecimal, '+', '-' and 'GO' key may be entered after
                 503  ; different function keys.  The monitor uses branch tables
                 504  ; and STATE to determine the current function and branch
                 505  ; to the proper entry point.
                 506  ;
                 507  KHEX:
                 508  ;Executed when hexadecimal keys are pressed.
                 509  ;Use HTAB and STATE for further branch.
                 510
0111   4F        511          LD      C,A       ;save A register in C
                 512                            ;which is the hex key-code.
0112   214B07    513          LD      HL,HTAB
0115   3AE41F    514  BR1     LD      A,(STATE)
0118   C3B003    515          JP      BRANCH
                 516  ;
                 517
                 518  KINC:
                 519  ;Branched by KSUBFUN table.
                 520  ;Executed when '+' key is pressed.
                 521  ;Use ITAB and STATE for further branch.
                 522  ;STATE is will be stored in A register at BR1.
                 523
011B   215707    524          LD      HL,ITAB
011E   18F5      525          JR      BR1
                 526
                 527  ;
                 528  KDEC:
                 529  ;Branched by KSUBFUN table.  Executed
                 530  ;when '-' key is pressed.  Use DTAB and
                 531  ;STATE for further branch.  STATE will be
                 532  ;stored in A register at BR1.
                 533
0120   216307    534          LD      HL,DTAB
0123   18F0      535          JR      BR1
                 536
                 537  ;
                 538  KGO:
                 539  ;Branched by KSUBFUN table.  Executed
                 540  ;when 'GO' key is pressed.  Use GTAB and
                 541  ;STATE for further branch.  STATE will be
                 542  ;stored in A register at BR1.
                 543
0125   216F07    544          LD      HL,GTAB
0128   18EB      545          JR      BR1
                 546
                 547  ;
                 548  KSTEP:
                 549  ;Branched by table KSUBFUN.  Executed
                 550  ;when 'STEP' key is pressed.
                 551
012A   CDE503    552          CALL    TESTM     ;Check if the left 4 digits
                 553                            ;of the display are memory address.
                 554                            ;If not, disable all LED's as
                 555                            ;a warning to the user.  This
                 556                            ;is done by routine IGNORE.
012D   C2BB03    557          JP      NZ,IGNORE
0130   3E80      558          LD      A,10000000B       ;This data will be output
                 559                                    ;to port B to enable
                 560                                    ;BREAK.  It is done by
                 561                                    ;routine PREOUT.
0132   C3A302    562          JP      PREOUT
                 563
                 564  ;
```

```
LOC    OBJ CODE M STMT SOURCE STATEMENT
                565 KDATA:
                566 ;Branched by table KSUBFUN.  Executed
                567 ;when 'DATA' key is pressed.
                568
0135   CDE503   569          CALL    TESTM     ;Check if the left 4 digits
                570                            ;of the display are memory address.
0138   2004     571          JR      NZ,TESTRG ;If not, branch to TESTRG
                572                            ;to check whether the display
                573                            ;is register or not.
013A   CD0B04   574          CALL    MEMDP2    ;If yes, display the data of
                575                            ;that address and set STATE
                576                            ;to 2.
013D   C9       577          RET
013E   FE08     578 TESTRG   CP      9         ;check if the status is 8 or 9
                579                            ;(RGAD or RGDA).
0140   DABB03   580          JP      C,IGNORE  ;If not, ignore this key and
                581                            ;send out a warning message.
0143   CD7704   582          CALL    REGDP9    ;If yes, display register and
                583                            ;set status to 9 (RGDA).
0146   C9       584          RET
                585
                586 ;
                587 KSBR:
                588 ;Branched by table KSUBFUN.  Executed
                589 ;when 'SBr' key (set break point) is
                590 ;pressed.
                591
0147   CDE503   592          CALL    TESTM     ;Check if the display is of
                593                            ;'address-data' form.
014A   C2BB03   594          JP      NZ,IGNORE ;If not, ignore this key and
                595                            ;send out a warning message.
014D   2ADE1F   596          LD      HL,(ADSAVE) ;If yes, get the address
                597                              ;being display now.
0150   CDF605   598          CALL    RAMCHK    ;Check if this address is
                599                            ;in RAM.
0153   C2BB03   600          JP      NZ,IGNORE ;If not, ignore the 'SBR' key
                601                            ;and send out a warning message.
0156   22E01F   602          LD      (BRAD),HL ;If yes, set this address as
                603                            ;a break point.
0159   CD0B04   604          CALL    MEMDP2    ;Display the data of break
                605                            ;address and set STATE to
                606                            ;2 (DA).
015C   C9       607          RET
                608
                609 ;
                610 KINS:
                611 ;Branched by table KSUBFUN.  Executed
                612 ;when 'Ins' key (insert) is pressed.
                613
015D   CDE503   614          CALL    TESTM     ;Check if the display is of
                615                            ;'address-data' form now.
0160   C2BB03   616          JP      NZ,IGNORE ;If not, ignore the 'INS' key
                617                            ;and send out a warning message.
0163   2ADE1F   618          LD      HL,(ADSAVE) ;If yes, get the address being
                619                              ;displayed now.
                620
0166   00       621          NOP
                622
0167   22AF1F   623          LD      (STEPBF),HL ; Store this address in
                624                              ;STEPBF and the next address
                625                              ;in STEPBF+4 for later use.
016A   23       626          INC     HL
016B   22B31F   627          LD      (STEPBF+4),HL
016E   CDF605   628          CALL    RAMCHK    ;Check if the address to be
                629                            ;inserted is in RAM.
0171   C2BB03   630          JP      NZ,IGNORE ;If not, ignore the 'INS' key
                631                            ;and send out a warning message.
                632                            ;If the address to be inserted
                633                            ;is in 1800-1DFF,store 1DFE into
                634                            ;STEPBF+2
                635                            ;Otherwise, ignore the 'INS' key.
                636                            ;This is done by the following
                637                            ;instructions.
0174   11FE1D   638          LD      DE,1DFEH
0177   7C       639          LD      A,H
0178   FE1E     640          CP      1EH
017A   3807     641          JR      C,SKIPH1
017C   FE20     642          CP      20H
017E   DABB03   643          JP      C,IGNORE
0181   1627     644          LD      D,27H
0183   ED53B11F 645 SKIPH1   LD      (STEPBF+2),DE
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 646
                 647  ;When one byte is inserted at some
                 648  ;address, all data below this address
                 649  ;will be shifted down one position.
                 650  ;The last location will be shifted out
                 651  ;and therefore lost.
                 652  ;The RAM is divided into 3 blocks as
                 653  ;insert is  concerned.  They are:
                 654  ;1800-1DFF,1E00-1FFF and 2000-27FF
                 655  ;The 2 nd block cannot be inserted and
                 656  ;is usually used as data bank.  System
                 657  ;data that of course cannot be shifted
                 658  ;are also stored in this bank.  Each
                 659  ;block is independent of the other when
                 660  ;shift is  performed, i.e. the data
                 661  ;shifted out of the first block will not
                 662  ;be propagated to next block.
                 663  ;The shift is accomplished by block
                 664  ;transfer, i.e. MOVE.  This is the
                 665  ;job of subroutine GMV.
                 666  ;Routine GMV needs 3 parameters which
                 667  ;are stored in step-buffer (STEPBF):
                 668  ;STEPBF: starting address (2 bytes);
                 669  ;STETBF+2: ending address (2 bytes);
                 670  ;STEPBF+4: destination address (2 bytes).
                 671
0187  CDE402     672  DOMV    CALL    GMV
018A  AF         673          XOR     A   ;After the RAM has been shifted down,
                 674                      ;the data of the address to be inserted
                 675                      ;is cleared to zero.  This is done by
                 676                      ;the next two instructions.  Register
                 677                      ;DE contain inserted address after GMV
                 678                      ;is performed.
018B  12         679          LD      (DE),A
018C  2AB31F     680          LD      HL,(STEPBF+4) ;Store the data in (STEPBF+4)
018F  22DE1F     681          LD      (ADSAVE),HL   ;into (ADSAVE).
0192  CD0B04     682          CALL    MEMDP2  ;Display the address and data, also
                 683                          ;set STATE to 2.
0195  C9         684          RET
                 685  ;
                 686  KDEL:
                 687  ;Branched by table KSUBFCN.  Executed
                 688  ;when 'Del' (delete) key is pressed.
                 689
0196  CDE502     690          CALL    TESIM   ;Check if the display is of
                 691                          ;'address-data' form.
0199  C2BB03     692          JP      NZ,IGNORE ;If not, ingore the 'Del' key and
                 693                            ;send out a warning message.
                 694                            ;'Delete' is quite similar to
                 695                            ;'Insert', except that the memory
                 696                            ;is shifted up instead of shifted
                 697                            ;down.  See the comments on
                 698                            ;routine EINS for detail.
019C  2ADE1F     699          LD      HL,(ADSAVE) ;Get the address being displayed
                 700                              ;now.  This is the address to
                 701                              ;be deleted.
                 702
                 703
019F  00         704          NOP
                 705
01A0  22B31F     706          LD      (STEPBF+4),HL
01A3  CDF605     707          CALL    RAMCHK  ;Check if the address is in RAM.
01A5  C2BB03     708          JP      NZ,IGNORE ;If not, ignore this key and
                 709                            ;send out a warning message.
                 710                          ;Following instructions prepare the
                 711                          ;parameters for routine GMV in step-
                 712                          ;buffer.  Refer to routine EINS for
                 713                          ;detail.
01A9  11001E     714          LD      DE,1E00H
01AC  7C         715          LD      A,H
01AD  FE1E       716          CP      1EH
01AF  3807       717          JR      C,SKIPH2
01B1  FE20       718          CP      20H
01B3  DABB03     719          JP      C,IGNORE
01B6  1628       720          LD      D,28H
01B8  ED53B11F   721  SKIPH2  LD      (STEPBF+2),DE
01BC  23         722          INC     HL
01BD  22AF1F     723          LD      (STEPBF),HL
01C0  18C5       724          JR      DOMV
                 725  ;
                 726  ;*****************************************************
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 727  KPC:
                 728  ; Branched by table KFUN.  Executed when
                 729  ; 'PC' key is pressed.
                 730
01C2  2ADC1F     731          LD      HL,(USERPC) ;Store the user's program
01C5  22DE1F     732          LD      (ADSAVE),HL ;counter into (ADSAVE)
01C8  CD0B04     733          CALL    MEMDP2  ;Routine MEMDP2 displays the address
                 734                          ;in (ADSAVE) and its data.  It also
                 735                          ;set the STATE to 2.
01CB  C9         736          RET
                 737  ;
                 738  KCBR:
                 739  ; Branched by table KFUN.  Executed when
                 740  ; 'CBr' (clear break point) key is pressed.
                 741
01CC  CDDE03     742          CALL    CLRBR   ;Call subroutine CBRBR to clear
                 743                          ;break point.  When returned, the HL
                 744                          ;register will contain FFFF.
01CF  22DE1F     745          LD      (ADSAVE),HL ;Store FFFF into (ADSAVE)
01D2  CD0B04     746          CALL    MEMDP2  ;Display address and its data.  Also
                 747                          ;set STATE to 2.
01D5  C9         748          RET
                 749  ;
                 750  KREG:
                 751  ; Branched by table KFUN.  Executed when
                 752  ; 'Reg' key is pressed.
01D6  DD21CA07   753          LD      IX,REG_ ;Routine SCAN uses IX as a pointer
                 754                          ;for display buffer.  Set IX to REG_.
                 755                          ;will make SCAN displays 'Reg-
01DA  CDC404     756          CALL    FCONV   ;Decode user's flag F and F' to
                 757                          ;binary display format.  This
                 758                          ;format will be used later, when
                 759                          ;user requires the monitor to
                 760                          ;display decoded flag by pressing
                 761                          ;keys 'SZXH', 'XPNC',...
01DD  C9         762          RET
                 763  ;
                 764  KADDR:
                 765  ; Branched by KFUN table.  Executed when
                 766  ; 'Addr' key is pressed.
                 767
01DE  CD0204     768          CALL    MEMDP1  ;Display the address stored in
                 769                          ;(ADSAVE) and its data.  Set STATE
                 770                          ;to 1 (AD).
01E1  C9         771          RET
                 772  ;
                 773  ; Function Move, Relative, Read-tape and
                 774  ; Write-tape require from one to three
                 775  ; parameters.  They are stored in STEPBF
                 776  ; (step buffer).  STMINOR (minor status)
                 777  ; contains the number of parameters has been
                 778  ; entered.  For Move and Relative, the
                 779  ; default value of the first parameter is
                 780  ; the address stored in (ADSAVE).  There
                 781  ; is no default value for the first parameter
                 782  ; (filename) of Read- and Write-tape.  When the
                 783  ; function keys are pressed, STMINOR is automatically
                 784  ; reset to 0.
                 785  ;
                 786
                 787  KMV:
                 788  ; Branched by table KFUN.  Executed when
                 789  ; 'Move' key is pressed.
                 790  KRL:
                 791  ; Branched by table KFUN.  Executed when
                 792  ; 'Rela' (relative) key is pressed.
01E2  2ADE1F     793          LD      HL,(ADSAVE) ;Store the contents of ADSAVE
                 794                              ;into STEPBF as default value
                 795                              ;of first parameter.
01E5  22AF1F     796          LD      (STEPBF),HL
                 797  KWT:
                 798  ; Branched by table KFUN.  Executed
                 799  ; when 'WRtape' key is pressed.
                 800
                 801  KRT:
                 802  ; Branched by table KFUN.  Executed when
                 803  ; 'RDtape' key is pressed.
                 804
01E8  CD3A04     805          CALL    STEPDP  ;Display the parameter that
                 806                          ;is being entered now by calling
                 807                          ;subroutine STEPDP.
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
01EB  C9          808          RET
                  809  ;
                  810  ;*****************************************************
                  811  ; The following subroutines with name H???
                  812  ; are the service routine for hexadecimal
                  813  ; keys corresponding to each STATE.  They
                  814  ;  are all branched by table HTAB and STATE.
                  815
01EC  C3BB03      816  HFIX    JP      IGNORE ;When the display is fixed pattern
                  817                         ;hexadecimal keys are illegal.
                  818                         ;Disable all LED's as a warning
                  819                         ;message to the user.  This is what
                  820                         ;routine IGNORE does.
                  821  ;
01EF  2ADE1F      822  HDA     LD      HL,(ADSAVE) ;Get the address being displayed
                  823                              ;now from (ADSAVE)
01F2  CDF605      824          CALL    RAMCHK  ;Check if it is in RAM.
01F5  C2BB03      825          JP      NZ,IGNORE ;If not, ignore this key and
                  826                              ;send out a warning message.
01F8  CDEE03      827          CALL    PRECL1  ;If this is the first hexadecimal
                  828                          ;key entered after function or sub-
                  829                          ;function key,reset the data of that
                  830                          ;address to 0. (by routine PERCL1)
01FB  79          831          LD      A,C     ;The key-code is saved in C at
                  832                          ;routine KHEX.  Restore it to A.
01FC  ED6F        833          RLD             ;Rotate the key-code (4 bits) into
                  834                          ;the address obtained above. (in HL)
01FE  CD0B04      835          CALL    MEMDP2  ;Display the address and data,
                  836                          ;then set STATE to 2 (DA).
0201  C9          837          RET
                  838  ;
0202  21DE1F      839  HAD:    LD      HL,ADSAVE
0205  CDFA03      840          CALL    PRECL2  ;If this is the first hezdecimal
                  841                          ;key after function key is entered,
                  842                          ;set the contents of ADSAVE to 0.
0208  79          843          LD      A,C     ;The key-code is saved in C
                  844                          ;by routine KHEX.
                  845                          ;The next three instructions shift
                  846                          ;the address being displayed  by
                  847                          ;one digit.
0209  ED6F        848          RLD
020B  23          849          INC     HL
020C  ED6F        850          RLD
020E  CD0204      851          CALL    MEMDP1  ;Display the address and its
                  852                          ;data.  Also, set STATE to 1.
0211  C9          853          RET
                  854  ;
                  855  HRGAD:
                  856  HRGFIX:
0212  79          857          LD      A,C
0213  DD21B61F    858          LD      IX,DISPBF
0217  21E31F      859          LD      HL,STMINOR
021A  87          860          ADD     A,A     ;The key-code is the register
                  861                          ;name.  Double it and store it
                  862                          ;into STMINOR.
021B  77          863          LD      (HL),A
021C  CD7304      864          CALL    REGDP8  ;Display register and set
                  865                          ;STATE to 8. (RGAD)
021F  C9          866          RET
                  867  ;
                  868  HRT:
                  869  HWT:
                  870  HRL:
0220  CD5504      871  HMV:    CALL    LOCSTBF ;Use STMINOR and STEPBF
                  872                          ;to calculate the address
                  873                          ;of current parameter in
                  874                          ;step buffer.
0223  CDFA03      875          CALL    PRECL2  ;If this is the first hex
                  876                          ;key entered, cleared the
                  877                          ;parameter (2 bytes) by
                  878                          ;PRECL2.
0226  79          879          LD      A,C     ;C contains the key-code.
                  880                          ;Rotate the parameter (2 bytes)
                  881                          ;1 digit left with the key-code.
0227  ED6F        882          RLD
0229  23          883          INC     HL
022A  ED6F        884          RLD
022C  CD3A04      885          CALL    STEPDP  ;Display the parameter.
022F  C9          886          RET
                  887  ;
0230  CDBB04      888  HRGDA   CALL    LOCRGBF ;Calculate the address of
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 889                                ;the register being modified.
0233  CDEE03     890          CALL    PRECL1        ;If this is the first hex
                 891                                ;key entered.  Clear the register
                 892                                ;(1 byte) by PRECL1.
0236  79         893          LD      A,C           ;Rotate user's register (1 byte)
                 894                                ;1 digit left with the key-code
                 895                                ;stored in C.
0237  ED6F       896          RLD
0239  CD7704     897          CALL    REGDP9        ;Display the register and set
                 898                                ;STATE to 9 (RGDA).
023C  C9         899          RET
                 900  ;
                 901  ;*****************************************************
                 902   ;The following routines with name
                 903   ;I???? are the service routines for
                 904   ;'+' key corresponding to each STATE.
                 905   ;They are all branched by table ITAB
                 906   ;and STATE.
                 907
                 908  IFIX:
                 909  IRGFIX:
023D  C3BB03     910          JP      IGNORE ;'+' key is illeagl for state
                 911                         ;FIX or RGFIX, ignore it.
                 912  ;
                 913  IAD:
0240  2ADE1F     914  IDA:    LD      HL,(ADSAVE) ;Increase the address being
                 915                              ;displayed now (in ADSAVE)
                 916                              ;by 1.
0243  23         917          INC     HL
0244  22DE1F     918          LD      (ADSAVE),HL
0247  CD0B04     919          CALL    MEMDP2        ;Display the address and data,
                 920                                ;then set the STATE to 2.
024A  C9         921          RET
                 922  ;
                 923  IRT:
                 924  IWT:
                 925  IRL:
024B  21E31F     926  IMV:    LD      HL,STMINOR ;STMINOR contains the
                 927                             ;parameter count, increment
                 928                             ;it by one.
024E  34         929          INC     (HL)
024F  CD5F04     930          CALL    LOCSTNA ;Check if the count is
                 931                          ;overflowed.
0252  2004       932          JR      NZ,ISTEP ;If not overflowed, continue
                 933                           ;at ISTEP.
0254  35         934          DEC     (HL)    ;Otherwise, restore the count
                 935                          ;and ignore the '+' key.
0255  C3BB03     936          JP      IGNORE
0258  CD3A04     937  ISTEP   CALL    STEPDP  ;Display the parameter at
                 938                          ,step buffer.
025B  C9         939          RET
                 940  ;
                 941  IRGAD:
025C  21E31F     942  IRGDA:  LD      HL,STMINOR ;In these states, the STMINOR
                 943                             ;contains the register name.
                 944                             ;Increase it by 1.  If it
                 945                             ;reaches the last one, reset
                 946                             ;it to the first one (0).
025F  34         947          INC     (HL)
0260  3E1F       948          LD      A,1FH
0262  BE         949          CP      (HL)
0263  3002       950          JR      NC,IRGNA
0265  3600       951          LD      (HL),0
0267  CD7704     952  IRGNA   CALL    REGDP9  ;Display the register and
                 953                          ;set STATE to 9.
026A  C9         954          RET
                 955  ;
                 956  ;*****************************************************
                 957   ;The following routines with name
                 958   ;D???? are the service routines for
                 959   ;'-' key corresponding to each state.
                 960   ;They are all branched by table DTAB
                 961   ;and STATE.
                 962
                 963  DFIX:
                 964  DRGFIX:
026B  C3BB03     965          JP      IGNORE ;'-' key is illegal for
                 966                         ;these states.  Ignore it.
                 967  ;
                 968  DAD:
026E  2ADE1F     969  DDA:    LD      HL,(ADSAVE) ;Decrease the address being
                 970                              ;displayed now (in ADSAVE)
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 971                                ;by one.
0271  2B         972          DEC     HL
0272  22DE1F     973          LD      (ADSAVE),HL
0275  CD0B04     974          CALL    MEMDP2   ;Display the address and data,
                 975                           ;set STATE to 2 (DA).
0278  C9         976          RET
                 977  ;
                 978  DRT:
                 979  DWT:
                 980  DRL:
0279  21E31F     981  DMV:    LD      HL,STMINOR ;In these states, STMINOR
                 982                             ;contains the parameter count.
                 983                             ;Decrease it by one.  If overflow
                 984                             ;occurs, restore STMINOR and
                 985                             ;ignore the '-' key.  Otherwise
                 986                             ;continue at DSTEP.
027C  35         987          DEC     (HL)
027D  CD5F04     988          CALL    LOCSTNA
0280  2004       989          JR      NZ,DSTEP
0282  34         990          INC     (HL)
0283  C3BB03     991          JP      IGNORE
0286  CD3A04     992  DSTEP   CALL    STEPDP   ;Display the parameter.
0289  C9         993          RET
                 994  ;
                 995  DRGAD:
028A  21E31F     996  DRGDA:  LD      HL,STMINOR ;In these states, STMINOR
                 997                             ;contains the register name.
                 998                             ;Decrease it by one.' If it
                 999                             ;goes below zero, set it to
                1000                             ;the highest value (1F).
028D  35        1001          DEC     (HL)
028E  3E1F      1002          LD      A,01FH
0290  BE        1003          CP      (HL)
0291  3002      1004          JR      NC,DRGNA
0293  361F      1005          LD      (HL),1FH
0295  CD7704    1006  DRGNA   CALL    REGDP9   ;Display the register and
                1007                           ;set STATE to 9.
0298  C9        1008          RET
                1009  ;
                1010  ;*****************************************************
                1011   ;The following routines with name
                1012   ;G???? are the service routines for
                1013   ;'GO' key  corresponding to each
                1014   ;state.  They are all branched by
                1015   ;table GTAB and STATE.
                1016
                1017  GFIX:
                1018  GRGFIX:
                1019  GRGAD:
0299  C3BB03    1020  GRGDA:  JP      IGNORE ;'GO' key is illegal for
                1021                         ;these states.  Ignore it.
                1022  ;
                1023  GAD:
029C  2AE01F    1024  GDA:    LD      HL,(BRAD) ;Get the address of break
                1025                            ;point.
029F  36EF      1026          LD      (HL),0EFH ;Instruction RST 28H.
                1027                            ;The content of break address
                1028                            ;is changed to RST 28H before
                1029                            ;the control is transfered to
                1030                            ;user's program.  This
                1031                            ;will cause a trap when user's
                1032                            ;PC passes this point.
02A1  3EFF      1033          LD      A,0FFH   ;Save FF into TEMP.  This data
                1034                           ;will be output to port B later.
                1035                           ;FF is used to disable break point.
02A3  32EA1F    1036  PREOUT  LD      (TEMP),A   ;Store A into TEMP.
02A6  3AD21F    1037          LD      A,(USERIF) ;Save two instructions into
                1038                             ;TEMP and TEMP+1.  These two
                1039                             ;instructions will be executed
                1040                             ;later.  If the user's IFF
                1041                             ;(interrupt flip-flop) is 1,
                1042                             ;the instructions are 'EI RET'.
                1043                             ;Otherwise, they are 'DI RET'.
02A9  CB47      1044          BIT     0,A
02AB  21FBC9    1045          LD      HL,0C9FBH        ;'EI','RET'
02AE  2002      1046          JR      NZ,EIDI
02B0  2EF3      1047          LD      L,0F3H           ;'DI'
02B2  22EB1F    1048  EIDI    LD      (TEMP+1),HL
02B5  31BC1F    1049          LD      SP,REGBF ;Restore user's registers by
                1050                           ;setting SP to REGBF (register
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 1051                                ;buffer) and continuously popping
                 1052                                ;the stack.
02B8  F1         1053          POP     AF
02B9  C1         1054          POP     BC
02BA  D1         1055          POP     DE
02BB  E1         1056          POP     HL
02BC  08         1057          EX      AF,AF'
02BD  F1         1058          POP     AF
02BE  08         1059          EX      AF,AF'
02BF  D9         1060          EXX
02C0  C1         1061          POP     BC
02C1  D1         1062          POP     DE
02C2  E1         1063          POP     HL
02C3  D9         1064          EXX
02C4  DDE1       1065          POP     IX
02C6  FDE1       1066          POP     IY
02C8  ED7BD01F   1067          LD      SP,(USERSP) ;Restore user's SP.
02CC  32BD1F     1068          LD      (USERAF+1),A ;Temporarily save A
02CF  3AD31F     1069          LD      A,(USERIF+1) ;Restore user's I
02D2  ED47       1070          LD      I,A
02D4  E5         1071          PUSH    HL          ;The next 3 instructions
                 1072                                ;push the address being
                 1073                                ;displayed now (in ADSAVE)
                 1074                                ;onto stack without changing
                 1075                                ;HL register.  This address will be
                 1076                                ;treated as user's new PC.
02D5  2ADE1F     1077          LD      HL,(ADSAVE)
02D8  E3         1078          EX      (SP),HL
02D9  3AEA1F     1079          LD      A,(TEMP) ;Output the data stored in
                 1080                               ;TEMP to port B of 8255.
                 1081                               ;This data is prepared by
                 1082                               ;routine KSTEP or GAD or
                 1083                               ;GDA.  In first case, it is
                 1084                               ;10111111 and will enable
                 1085                               ;break point.  In other
                 1086                               ;cases, it is FF and will
                 1087                               ;disable break point.
                 1088                               ;If break is enabled, non-
                 1089                               ;maskable interrupt will occur
                 1090                               ;5 M1's after the OUT instruction.
02DC  D302       1091          OUT     (DIGIT),A
02DE  3ABD1F     1092          LD      A,(USERAF+1) ;1st M1,
                 1093                                ;Restore A register.
02E1  C3EB1F     1094          JP      TEMP+1  ;2nd M1,
                 1095                          ;Execute the two instructions
                 1096                          ;stored in RAM.  They are:
                 1097                          ;    EI (or DI)      ;3rd M1
                 1098                          ;    RET             ;4th M1
                 1099                          ;The starting address of user's
                 1100                          ;program has been pushed onto
                 1101                          ;the top of the stack.  RET pops
                 1102                          ;out this address and transfers
                 1103                          ;control to it.  The first M1
                 1104                          ;of user's program will be the
                 1105                          ;5th M1 after OUT. If break point
                 1106                          ;is enabled, NMI will occur after
                 1107                          ;this instruction is completed.
                 1108                          ;This is the mechanism of single
                 1109                          ;step.
                 1110  ;
                 1111  ;*************************************************************
02E4  21AF1F     1112  GMV     LD      HL,STEPBF
02E7  CD3D05     1113          CALL    GETP    ;Load parameters from
                 1114                          ;step buffer into registers.
                 1115                          ;Also check if the parameters
                 1116                          ;are legal.  After GETP,
                 1117                          ;HL = start address of source
                 1118                          ;BC = length to MOVE.
02EA  3867       1119          JR      C,ERROR ;Jump to ERROR if the
                 1120                          ;parameters are illegal. (I.e., Ending
                 1121                          ;address < starting address.)
02EC  ED5BB31F   1122          LD      DE,(STEPBF+4) ;Load destination
                 1123                                ;address into DE.
02F0  ED52       1124          SBC     HL,DE   ;Compare HL and DE to
                 1125                          ;determine move up or down.
02F2  300C       1126          JR      NC,MVUP
                 1127                          ;Move down:
02F4  EB         1128          EX      DE,HL   ;HL = destination address
02F5  09         1129          ADD     HL,BC   ;HL = dest. address + length
02F6  2B         1130          DEC     HL      ;HL = end address of dest.
02F7  EB         1131          EX      DE,HL   ;DE = end address of dest.
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
02F8  2AB11F     1132          LD      HL,(STEPBF+2) ;HL = end address of source
02FB  EDB8       1133          LDDR                  ;block transfer instruction
02FD  13         1134          INC     DE            ;DE = last address moved
02FE  181C       1135          JR      ENDFUN        ;Continue at ENDFUN.
                 1136  MVUP:                          ;Move up:
0300  19         1137          ADD     HL,DE         ;HL is destroyed by
                 1138                                ;SBC HL,DE.  Restore HL.
0301  EDB0       1139          LDIR                  ;block transfer
0303  1B         1140          DEC     DE            ;DE = last address moved
0304  1816       1141          JR      ENDFUN        ;Continue at ENDFUN.
                 1142  ;
                 1143  ;*************************************************************
0306  ED5BAF1F   1144  GRL     LD      DE,(STEPBF)   ;Load starting address
                 1145                                ;into DE.
030A  13         1146          INC     DE            ;Increase this address by 2.
                 1147                                ;Relative address is used in
                 1148                                ;instruction JR or DJNZ.
                 1149                                ;The codes for them are 2 bytes.
                 1150                                ;The PC is increased by 2 after
                 1151                                ;opcode is fetched.
030B  13         1152          INC     DE
030C  2AB11F     1153          LD      HL,(STEPBF+2) ;Load destionation
                 1154                                ;address into HL.
030F  B7         1155          OR      A
0310  ED52       1156          SBC     HL,DE         ;Calculate difference.
0312  7D         1157          LD      A,L           ;Check if the offset is between
                 1158                                ;+127 (007FH) and -128 (FF80H).
                 1159                                ;If the offset is positive, both H
                 1160                                ;and bit 7 of L must be zero; if it
                 1161                                ;is negative, H and bit 7 of L must
                 1162                                ;be FF and 1.  In both cases, adding
                 1163                                ;H with bit 7 of L results in 0.
0313  17         1164          RLA                   ;Rotate bit 7 of L into carry flag.
0314  7C         1165          LD      A,H
0315  CE00       1166          ADC     A,0           ;ADD H and bit 7 of L.
0317  203A       1167          JR      NZ,ERROR      ;Branch to ERROR if
                 1168                                ;the result is nonzero.
0319  7D         1169          LD      A,L
031A  1B         1170          DEC     DE
031B  12         1171          LD      (DE),A        ;Save the offset into
                 1172                                ;the next byte of opcode.
                 1173                                ;(DJNZ or JR)
                 1174  ;
                 1175  ENDFUN:
031C  ED53DE1F   1176          LD      (ADSAVE),DE   ;Save DE into ADSAVE.
0320  CD0B04     1177          CALL    MEMDP2        ;Display this address and
                 1178                                ;its data.  Set STATE to 2.
0323  C9         1179          RET
                 1180  ;
                 1181  ;*************************************************************
                 1182  GWT:
0324  CD2D05     1183          CALL    SUM1          ;Load parameters from
                 1184                                ;step buffer into registers.
                 1185                                ;Check if the parameters
                 1186                                ;are legal.  If legal, calculate
                 1187                                ;the sum of all data to be output
                 1188                                ;to tape.
0327  382A       1189          JR      C,ERROR       ;Branch to ERROR if the
                 1190                                ;parameters are illegal. (lenght is
                 1191                                ;negative)
0329  32B51F     1192          LD      (STEPBF+6),A  ;Store the checksum into
                 1193                                ;STEPBF+6.
032C  21A00F     1194          LD      HL,4000       ;Output 1k Hz square
                 1195                                ;wave for 4000 cycles.
                 1196                                ;Leading sync. signal.
032F  CDDE05     1197          CALL    TONE1K
0332  21AF1F     1198          LD      HL,STEPBF     ;Output 7 bytes starting
                 1199                                ;at STEPBF. (Include:
                 1200                                ;filename, starting, ending
                 1201                                ;address and checksum)
0335  010700     1202          LD      BC,7
0338  CDA705     1203          CALL    TAPEOUT
033B  21A00F     1204          LD      HL,4000       ;Output 2k Hz square
                 1205                                ;wave for 4000 cycles.
                 1206                                ;Middle sync.  The file name of the
                 1207                                ;file being read will be displayed
                 1208                                ;in this interval.
033E  CDE205     1209          CALL    TONE2K
0341  CD3A05     1210          CALL    GETPTR        ;Load parameters into
                 1211                                ;registers. (Starting, ending and
                 1212                                ;length).
0344  CDA705     1213          CALL    TAPEOUT       ;Output user's data.
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
0347  21A00F     1214          LD      HL,4000 ;Output 4000 cycles of
                  1215                          ;2k Hz squire wave.
                  1216                          ;(Tail sync.)
034A  CDE205      1217          CALL    TONE2K
034D  ED5BB31F    1218  ENDTAPE LD      DE,(STEPBF+4) ;DE = last address
0351  18C9        1219          JR      ENDFUN  ;Continue at ENDFUN.
                  1220  ;
0353  DD21A907    1221  ERROR   LD      IX,ERR_ ;IX points to '-Err
0357  C3D000      1222          JP      SETSTO  ;Set STATE to 0 by
                  1223                          ;branching to SETSTO.
                  1224  ;
                  1225  ;*****************************************************
                  1226  GRT:
035A  2AAF1F      1227          LD      HL,(STEPBF) ;Temporarily save filename.
035D  22EA1F      1228          LD      (TEMP),HL
0360  3E40        1229  LEAD    LD      A,01000000B     ;decimal point
0362  D301        1230          OUT     (SEG7),A ;When searching for filename,
                  1231                          ;the display is blank  initially.
                  1232                          ;If the data read from MIC is
                  1233                          ;acceptable 0 or 1, the display
                  1234                          ;becomes '......'.
0364  21E803      1235          LD      HL,1000
0367  CD8C05      1236  LEAD1   CALL    PERIOD ;The return of PERIOD
                  1237                         ;is in flag:
                  1238                         ;  NC -- tape input is 1k Hz;
                  1239                         ;   C -- otherwise.
036A  38F4        1240          JR      C,LEAD ;Loop until leading sync.
                  1241                         ;is detected.
036C  2B          1242          DEC     HL      ;Decrease HL by one when
                  1243                          ;one period is detected.
036D  7C          1244          LD      A,H
036E  B5          1245          OR      L       ;Check if both H and L are 0.
036F  20F6        1246          JR      NZ,LEAD1 ;Wait for 1000 periods.
                  1247                          ;The leading sync. is accepted
                  1248                          ;if it is longer than 1000
                  1249                          ;cycles (1 second).
0371  CD8C05      1250  LEAD2   CALL    PERIOD
0374  30FB        1251          JR      NC,LEAD2 ;Wait all leading sync. to
                  1252                          ;pass over.
                  1253
0376  21AF1F      1254          LD      HL,STEPBF ;Load 7 bytes from
                  1255                          ;tape into STEPBF.
0379  010700      1256          LD      BC,7
037C  CD4D05      1257          CALL    TAPEIN
037F  38DF        1258          JR      C,LEAD  ;Jump to LEAD if input
                  1259                          ;is not successful.
0381  ED5BAF1F    1260          LD      DE,(STEPBF) ;Get filename from
                  1261                          ;step buffer.
0385  CD6506      1262          CALL    ADDRDP  ;Convert it to display
                  1263                          ;format.
0388  0696        1264          LD      B,150   ;Display it for 1.5 sec.
038A  CD2406      1265  FILEDP  CALL    SCAN1
038D  10FB        1266          DJNZ    FILEDP
038F  2AEA1F      1267          LD      HL,(TEMP) ;Check if the input
                  1268                          ;filename equals to the
                  1269                          ;specified filename.
0392  B7          1270          OR      A
0393  ED52        1271          SBC     HL,DE
0395  20C9        1272          JR      NZ,LEAD ;If not, find the leading
                  1273                          ;sync. of next file.
                  1274
                  1275                          ;If filename is found,
0397  3E02        1276          LD      A,00000010B     ;segament '-'
0399  D301        1277          OUT     (SEG7),A ;Display '------'.
039B  CD3A05      1278          CALL    GETPTR  ;The parameters (starting
                  1279                          ;ending address and check-
                  1280                          ;sum) have been load into
                  1281                          ;STEPBF.  Load them into
                  1282                          ;registers, calculate the block
                  1283                          ;length and check if they are
                  1284                          ;legal.
039E  38B3        1285          JR      C,ERROR ;Jump to ERROR if the
                  1286                          ;parameters are illegal.
03A0  CD4D05      1287          CALL    TAPEIN  ;Input user's data.
03A3  38AE        1288          JR      C,ERROR ;Jump to ERROR if input
                  1289                          ;is not successful.
03A5  CD2D05      1290          CALL    SUM1    ;Calculate the sum of all
                  1291                          ;input data.
03A8  21B51F      1292          LD      HL,STEPBF+6
03AB  BE          1293          CP      (HL)    ;Compare it with the
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 1294                          ;checksum calculated by and stored
                 1295                          ;'WRtape'.
03AC  20A5       1296        JR      NZ,ERROR  ;Jump to ERROR if not
                 1297                          ;matched.
03AE  189D       1298        JR      ENDTAPE   ;Continue at ENDTAPE.
                 1299  ;
                 1300  ;*****************************************************
                 1301  BRANCH:
                 1302  ;Branch table format:
                 1303  ;   byte 1,2 : address of the 1st routine in
                 1304  ;              each group.
                 1305  ;   byte 3   : difference between the address
                 1306  ;              of 1st and 1st routine, which is
                 1307  ;              of course 0.
                 1308  ;   byte 4   : difference between the address
                 1309  ;              of 2nd and 1st routine
                 1310  ;   byte 5   : difference between the address
                 1311  ;              of 3rd and 1st routine
                 1312  ;    ...
                 1313  ;    ...
                 1314  ;    ...
                 1315  ; HL : address of branch table
                 1316  ; A  : the routine number in its group
                 1317  ; Such branch table can save table length and avoid page
                 1318  ; (256 bytes) boundary problem.
                 1319
03B0  5E         1320        LD      E,(HL)    ;Load the address of 1st
                 1321                          ;routine in the group into
                 1322                          ;DE register.
03B1  23         1323        INC     HL
03B2  56         1324        LD      D,(HL)
03B3  23         1325        INC     HL        ;Locate the pointer of difference
                 1326                          ;table.
03B4  85         1327        ADD     A,L
03B5  6F         1328        LD      L,A
03B6  6E         1329        LD      L,(HL)    ;Load the address
                 1330                          ;difference into L.
03B7  2600       1331        LD      H,0
03B9  19         1332        ADD     HL,DE     ;Get routine's real address
03BA  E9         1333        JP      (HL)      ;Jump to it.
                 1334  :
                 1335  ;*****************************************************
                 1336  IGNORE:
03BB  21E61F     1337        LD      HL,TEST
03BE  CBFE       1338        SET     7,(HL)    ;Routine SCAN will check bit
                 1339                          ;7 of TEST.  If it is set,
                 1340                          ;all LEDs will be disabled.
                 1341                          ;This is a warning message to
                 1342                          ;the user when a illegal key
                 1343                          ;is entered.
03C0  C9         1344        RET
                 1345  ;
                 1346  ;*****************************************************
                 1347  INI:
                 1348  ; Power-up initialization.
03C1  DD21A507   1349        LD      IX,BLANK  ;BLANK is the initial pattern
                 1350
                 1351                          ;Display the following
                 1352  ——                      ;patterns sequence, each 0.16
                 1353                          ;seconds:
                 1354                          ;    '      '
                 1355                          ;    '     u'
                 1356                          ;    '    uP'
                 1357                          ;    '   uPF'
                 1358                          ;    '  uPF-'
                 1359                          ;    ' uPF--'
                 1360                          ;    'uPF--1'
                 1361
03C5  0E07       1362        LD      C,7       ;pattern count
03C7  0610       1363  INI1  LD      B,10H     ;Display 0.16 second.
03C9  CD2406     1364  INI2  CALL    SCAN1
03CC  10FB       1365        DJNZ    INI2
03CE  DD2B       1366        DEC     IX        ;next pattern
03D0  0D         1367        DEC     C
03D1  20F4       1368        JR      NZ,INI1
                 1369  ;
03D3  3EA5       1370        LD      A,PWCODE
03D5  C3B306     1371        JP      INI3
03D8  216600     1372  INI4  LD      HL,NMI
03DB  22EE1F     1373        LD      (IM1AD),HL ;Set the service routine
                 1374                           ;of RST 38H to NMI, which is the
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 1375                                ;nonmaskable interrupt service
                 1376                                ;routine for break point and
                 1377                                ;single step.
                 1378  CLRBR:
                 1379  ; Clear break point by setting
                 1380  ; the break point address to
                 1381  ; FFFF.  This is a non-existant
                 1382  ; address, so break can never
                 1383  ; happen.
                 1384
03DE   21FFFF    1385          LD      HL,0FFFFH
03E1   22E01F    1386          LD      (BRAD),HL
03E4   C9        1387          RET
                 1388  ;
                 1389  TESTM:
                 1390  ; Check if the display is of 'address-data'
                 1391  ; form, i.e. STATE 1 or 2.
                 1392  ; The result is stored in zero flag.
                 1393  ;   Z:  yes
                 1394  ;  NZ:  no
                 1395
03E5   3AE41F    1396          LD      A,(STATE)
03E8   FE01      1397          CP      1
03EA   C8        1398          RET     Z
03EB   FE02      1399          CP      2
03ED   C9        1400          RET
                 1401  ;
                 1402  PRECL1:
                 1403  ; Pre-clear 1 byte.
                 1404  ; If bit 0 of TEST is not 0, load 0 into (HL).  Bit 0 of
                 1405  ; TEST is cleared after check.
                 1406  ; Only AF register are destroyed.
                 1407
03EE   3AE61F    1408          LD      A,(TEST)
03F1   B7        1409          CR      A        ;Is bit 0 of TEST zero?
03F2   C8        1410          RET     Z
03F3   3E00      1411          LD      A,0
03F5   77        1412          LD      (HL),A   ;Clear (HL)
03F6   32E61F    1413          LD      (TEST),A ;Clear TEST too.
03F9   C9        1414          RET
                 1415  ;
                 1416  PRECL2:
                 1417  ; Pre-clear 2 bytes.
                 1418  ; If bit 0 of TEST is nonzero, clear (HL)
                 1419  ; and (HL+1).
                 1420  ; Only AF register are destroyed.
                 1421
03FA   CDEE03    1422          CALL    PRECL1
03FD   C8        1423          RET     Z
03FE   23        1424          INC     HL
03FF   77        1425          LD      (HL),A
0400   2B        1426          DEC     HL
0401   C9        1427          RET
                 1428  ;
                 1429  ;*************************************************************
                 1430  ; Memory display format: (address-data)
                 1431
                 1432  ;      i) A.A.A.A. D D -- State is AD.  four decimal points
                 1433  ;                         under the address field indicate
                 1434  ;                         that the numeric key entered will
                 1435  ;                         be interpreted as memory address.
                 1436  ;     ii) A A A A  D.D.-- State is DA.  Two decimal points
                 1437  ;                         under the data field indicate
                 1438  ;                         the monitor is expecting user to
                 1439  ;                         enter memory data.
                 1440  ;    iii) A.A.A.A. D.D.-- Six decimal points indicate the
                 1441  ;                         address being displayed is set
                 1442  ;                         as a break point.
                 1443
                 1444  MEMDP1:
0402   3E01      1445          LD      A,1      ;Next STATE =1
0404   0604      1446          LD      B,4      ;4 decimal points active
0406   21B81F    1447          LD      HL,DISPBF+2 ;The first active decimal
                 1448                           ;point is in DISPBF+2, the
                 1449                           ;last in DESPBF+5.
0409   1807      1450          JR      SAV12    ;Continue at SAV12.
                 1451  MEMDP2:
040B   3E02      1452          LD      A,2      ;Next STATE = 2
040D   0602      1453          LD      B,2      ;2 active decimal points
040F   21B61F    1454          LD      HL,DISPBF ;1st decimal point is in
                 1455                           ;DISPBF, 2nd in DISPBF+1.
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
0412  32E41F     1456  SAV12   LD      (STATE),A ;Update STATE
0415  D9         1457          EXX             ;Save register HL,BC,DE.
0416  ED5BDE1F   1458          LD      DE,(ADSAVE) ;The address to be
                 1459                            ;displayed is stored in
                 1460                            ;(ADSAVE).  Load it into
                 1461                            ;DE register.
041A  CD6506     1462          CALL    ADDRDP  ;Convert this address to
                 1463                          ;display format and store it
                 1464                          ;into DISPBF+2 ¢ DISPBF+5.
041D  1A         1465          LD      A,(DE) ;Load the data of this
                 1466                         ;address into A register.
041E  CD7106     1467          CALL    DATADP ;Convert this data to
                 1468                         ;display format and store it
                 1469                         ;into DISPBF ¢ DISPBF+1.
                 1470  BRTEST:
                 1471  ; The next 3 instructions serve to refresh the
                 1472  ; data at break address every time memory is
                 1473  ; displayed.
0421  2AE01F     1474          LD      HL,(BRAD) ;Get break point address.
0424  7E         1475          LD      A,(HL)  ;Get the data of this
                 1476                          ;address into A register.
0425  32E21F     1477          LD      (BRDA),A ;Store it into BRDA (break data).
0428  B7         1478          OR      A
0429  ED52       1479          SBC     HL,DE   ;Check if the address to
                 1480                          ;be displayed is break point.
042B  2006       1481          JR      NZ,SETPT1 ;If not, jump to SETPT1.
042D  0606       1482          LD      B,6     ;6 active decimal points.
042F  21B61F     1483          LD      HL,DISPBF ;1st decimal point is in
                 1484                            ;DISPBF; 6th in DISPBF+5.
0432  D9         1485          EXX
0433  D9         1486  SETPT1  EXX             ;Restore HL,BC,DE.
0434  CBF6       1487  SETPT   SET     6,(HL) ;Set decimal points.
                 1488                         ;Count in B, first address
                 1489                         ;in HL register.
0436  23         1490          INC     HL
0437  10FB       1491          DJNZ    SETPT
0439  C9         1492          RET
                 1493  ;
                 1494  ;*********************************************************
                 1495  ; Step display format: (this format is used when user is
                 1496  ; entering parameters for Move, Rela, WRtape, RDtape.)
                 1497
                 1498  ;          P.P.P.P. - N
                 1499
                 1500  ; 'P' is the digit of parameter.  Four decimal points
                 1501  ; indicate P's are being modified now.  N is the mnemonic of
                 1502  ; the parameter:
                 1503  ;      i) Move    S -- starting address
                 1504  ;                 E -- ending address
                 1505  ;                 D -- destination address
                 1506  ;     ii) Rela    S -- source address
                 1507  ;                 D -- destination address
                 1508  ;    iii) WRtape  F -- file name
                 1509  ;                 S -- starting address
                 1510  ;                 E -- ending address
                 1511  ;     iv) RDtape  F -- file name
                 1512
                 1513  STEPDP:
                 1514  ;Display step buffer and its parameter name.
                 1515  ;Input: STATE
                 1516  ;       STMIONR (parameter count)
                 1517  ;register destroyed: AF,BC,DE,HL
                 1518
043A  CD5504     1519          CALL    LOCSTBF ;Get parameter address
043D  5E         1520          LD      E,(HL)  ;Load parameter into DE
043E  23         1521          INC     HL
043F  56         1522          LD      D,(HL)
0440  CD6506     1523          CALL    ADDRDP  ;Convert this parameter to
                 1524                          ;display format (4 digits)
                 1525                          ;and store it into DISPBF+2
                 1526                          ; ¢ DISPBF+5.
0443  21B81F     1527          LD      HL,DISPBF+2 ;Set 4 decimal points.
                 1528                              ;From DISPBF+2 to DISPBF+5.
0446  0604       1529          LD      B,4
0448  CD3404     1530          CALL    SETPT
044B  CD5F04     1531          CALL    LOCSTNA ;Get parameter name.
044E  6F         1532          LD      L,A
044F  2602       1533          LD      H,2      ;Pattern '-' for 2nd rightmost
                 1534                           ;digit.
0451  22B61F     1535          LD      (DISPBF),EL
0454  C9         1536          RET
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 1537  ;
                 1538  LOCSTBF:
                 1539  ;Get the location of parameter.
                 1540  ; address = STEPBF + STMINOR*2
                 1541  ;register destroyed: AF,HL
                 1542
0455  3AE31F     1543          LD      A,(STMINOR) ;Get parameter count.
0458  87         1544          ADD     A,A         ;Each parameter has 2 bytes.
0459  21AF1F     1545          LD      HL,STEPBF   ;Get base address.
045C  85         1546          ADD     A,L
045D  6F         1547          LD      L,A
045E  C9         1548          RET
                 1549  ;
                 1550  LOCSTNA:
                 1551   ;Get parameter name.
                 1552   ;Input: STATE, STMINOR
                 1553   ;Output: parameter name in A, and Z flag.
                 1554
                 1555   ;register destroyed: AF,DE
045F  3AE41F     1556          LD      A,(STATE) ;Get STATE.
                 1557                            ;Possible states are:
                 1558                            ;4,5,6,7. (Move, Rel,
                 1559                            ;WRtape, RDtape)
0462  D604       1560          SUB     4         ;Change 4,5,6,7 to
                 1561                            ;0,1,2,3.
0464  87         1562          ADD     A,A     ;Each state has 4 bytes for names.
0465  87         1563          ADD     A,A
0466  11BC07     1564          LD      DE,STEPTAB
0469  83         1565          ADD     A,E
046A  5F         1566          LD      E,A     ;Now, DE contains the
                 1567                          ;address of 1st name
                 1568                          ;for each state.
046B  3AE31F     1569          LD      A,(STMINOR) ;Get parameter count
046E  83         1570          ADD     A,E         ;DE <--- DE + A
046F  5F         1571          LD      E,A
0470  1A         1572          LD      A,(DE)  ;Get parameter name.
0471  B7         1573          OR      A       ;Change zero flag.  If the
                 1574                          ;returned pattern (in A) is
                 1575                          ;zero,  the '+' or '-' must
                 1576                          ;have been pressed beyond legal
                 1577                          ; parameter boundary. (Check if
                 1578                          ;parameter name got from STEPTAB
                 1579                          ;is zero)
0472  C9         1580          RET
                 1581  ;
                 1582  ;*****************************************************
                 1583  ; Register display format:
                 1584
                 1585  ;       i)  X X X X  Y Y -- State is REGAD. The numeric data
                 1586  ;                           entered is interpreted as
                 1587  ;                           register name.
                 1588  ;                           YY is the register name, the
                 1589  ;                           data of that register pair is
                 1590  ;                           XXXX.
                 1591
                 1592  ;      ii)  X X X.X. Y Y  or
                 1593  ;     iii)  X.X.X X  Y Y -- State is REGDA.  The unit of
                 1594  ;                           register modification is byte.
                 1595  ;                           The numeric data entered will
                 1596  ;                           change the byte with decimal
                 1597  ;                           points under it.  Decimal points
                 1598  ;                           can be moved by '+' of '-' keys.
                 1599
                 1600  REGDP8:
                 1601  ; Display register and set STATE to 8.
                 1602
0473  3E08       1603          LD      A,8     ;Next state = 8
0475  1802       1604          JR      RGSTIN
                 1605
                 1606  REGDP9:
                 1607  ; Display register and set STATE to 9.
                 1608
0477  3E09       1609          LD      A,9     ;Next state = 9
                 1610
                 1611  RGSTIN:
                 1612  ; Update STATE by register A.
                 1613  ; Display user's register (count
                 1614  ; contained in STMINOR).
                 1615  ; register destroyed: AF,BC,DE,HL
                 1616
0479  32E41F     1617          LD      (STATE),A ;Update STATE.
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
047C  3AE31F     1618          LD      A,(STMINOR) ;Get register count.
047F  CB87       1619          RES     0,A         ;Registers are displayed by
                 1620                                ;pair.  Find the count
                 1621                                ;of pair leader.  (count of
                 1622                                ;the lower one)
0481  47         1623          LD      B,A         ;Temporarily save A.
0482  CDAE04     1624          CALL    RGNADP      ;Find register count.
                 1625                                ;Store them into DISPBF
                 1626                                ;and DISPBF+1.
0485  78         1627          LD      A,B         ;Restore A (register pair leader).
0486  CDBE04     1628          CALL    LOCRG       ;Get the address of
                 1629                                ;user's register.
0489  5E         1630          LD      E,(HL)      ;Get register data. (2 bytes)
048A  23         1631          INC     HL
048B  56         1632          LD      D,(HL)
048C  ED53DE1F   1633          LD      (ADSAVE),DE ;Convert them to display
                 1634                                ;format and store into
                 1635                                ;display buffer.
0490  CD6506     1636          CALL    ADDRDP
0493  3AE41F     1637          LD      A,(STATE)
0496  FE09       1638          CP      9           ;If STATE equals to 9 (RGDA),
                 1639                                ;set 2 decimal points.
                 1640                                ;Otherwise return here.
0498  C0         1641          RET     NZ
0499  21B81F     1642          LD      HL,DISPBF+2
049C  3AE31F     1643          LD      A,(STMINOR) ;Get register name.
049F  CB47       1644          BIT     0,A         ;If this register is
                 1645                                ;group leader, set decimal
                 1646                                ;points of two central digits.
                 1647                                ;Otherwise set two left digits.
04A1  2802       1648          JR      Z,LOCPT
04A3  23         1649          INC     HL
04A4  23         1650          INC     HL
04A5  CBF6       1651  LOCPT   SET     6,(HL)      ;Set decimal points of
                 1652                              ;(HL) and (HL+1)
04A7  23         1653          INC     HL
04A8  CBF6       1654          SET     6,(HL)
04AA  CDC404     1655          CALL    FCONV       ;Convert user's flag (F,F')
                 1656                                ;to binary display format.
04AD  C9         1657          RET
                 1658  ;
                 1659  RGNADP:
                 1660  ; Get the patterns of register names and
                 1661  ; store them into DISPBF and DISPBF+1.
                 1662  ; Input: A contains register count of
                 1663  ;        pair leader.
                 1664  ; register destroyed: AF,DE,HL
                 1665
04AE  21D007     1666          LD      HL,RGTAB    ;Get address of pattern
                 1667                                ;table.
04B1  85         1668          ADD     A,L
04B2  6F         1669          LD      L,A
04B3  5E         1670          LD      E,(HL)      ;Get first pattern.
04B4  23         1671          INC     HL
04B5  56         1672          LD      D,(HL)      ;Get 2nd pattern.
04B6  ED53B61F   1673          LD      (DISPBF),DE
04BA  C9         1674          RET
                 1675  ;
                 1676  LOCRGBF:
                 1677  ; Get the address of user's register.
                 1678  ; Register name contained in STMINOR.
                 1679  ; Destroys HL, AF.
                 1680
04BB  3AE31F     1681          LD      A,(STMINOR)
04BE  21BC1F     1682  LOCRG   LD      HL,REGBF
04C1  85         1683          ADD     A,L
04C2  6F         1684          LD      L,A
04C3  C9         1685          RET
                 1686  ;
                 1687  FCONV:
                 1688  ; Encode or decode user's flag register.
                 1689  ; STMINOR contains the name of the flag
                 1690  ; being displayed now.
                 1691  ; register destroyed: AF,BC,HL.
                 1692
04C4  3AE31F     1693          LD      A,(STMINOR) ;Get register name.
04C7  B7         1694          OR      A           ;Clear carry flag.
04C8  1F         1695          RRA                 ;name of I register: 17H,
                 1696                              ;name of IFF: 16H.
                 1697                              ;Rotate right one bit, both
                 1698                              ;become 0BH.
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
04C9  FE0B       1699          CP      0BH
04CB  2809       1700          JR      Z,FLAGX ;Jump to FLAGX if
                 1701                          ;I or IFF is being
                 1702                          ;displayed now.
04CD  4F         1703          LD      C,A     ;Otherwise, mask out bit
                 1704                          ;1 to bit 7 of user's IFF.
                 1705                          ;IFF is only 1 bit, monitor
                 1706                          ;use one byte to store it,
                 1707                          ;masking out bit 1¢7 is to
                 1708                          ;ignore the useless bits.
                 1709                          ;This is done only when the
                 1710                          ;user is not modifying IFF.
                 1711                          ;If user is modifying IFF,
                 1712                          ;monitor will display whatever
                 1713                          ;he enters, even if bit 1¢7
                 1714                          ;are not all zero.
                 1715                          ;A register is not changed
                 1716                          ;after doing this.
04CE  21D21F     1717          LD      HL,USERIF
04D1  7E         1718          LD      A,(HL)
04D2  E601       1719          AND     00000001B
04D4  77         1720          LD      (HL),A
04D5  79         1721          LD      A,C
04D6  FE0C       1722  FLAGX   CP      0CH     ;If STMINOR contains
                 1723                          ;the name of SZXH, XPNC,
                 1724                          ;SZXH' or XPNC', after
                 1725                          ;rotating right one bit
                 1726                          ;it will be greater than
                 1727                          ;or equal to 0CH.
                 1728                          ;Decode user's flag if it
                 1729                          ;is not being modified now,
                 1730                          ;encode it otherwise.
04D8  301F       1731          JR      NC,FCONV2
04DA  3ABC1F     1732  FCONV1  LD      A,(USERAF) ;Get user's F register.
04DD  CD1805     1733          CALL    DECODE  ;Decode upper 4 bits.
04E0  22D41F     1734          LD      (FLAGH),HL
04E3  CD1805     1735          CALL    DECODE  ;Decode lower 4 bits.
04E6  22D61F     1736          LD      (FLAGL),HL
04E9  3AC41F     1737          LD      A,(UAFP) ;Get user's F' register.
04EC  CD1805     1738          CALL    DECODE
04EF  22D81F     1739          LD      (FLAGHP),HL
04F2  CD1805     1740          CALL    DECODE
04F5  22DA1F     1741          LD      (FLAGLP),HL
04F8  C9         1742          RET
04F9  2AD41F     1743  FCONV2  LD      HL,(FLAGH) ;Get the binary form
                 1744                             ;of 4 upper bits of
                 1745                             ;user's F register.
04FC  CD2305     1746          CALL    ENCODE     ;Encode it.
04FF  2AD61F     1747          LD      HL,(FLAGL) ;Encode 4 lower bits.
0502  CD2305     1748          CALL    ENCODE
0505  32BC1F     1749          LD      (USERAF),A ;Save the encoded
                 1750                             ;result into USERAF.
0508  2AD81F     1751          LD      HL,(FLAGHP) ;Encode F' register.
050B  CD2305     1752          CALL    ENCODE
050E  2ADA1F     1753          LD      HL,(FLAGLP)
0511  CD2305     1754          CALL    ENCODE
0514  32C41F     1755          LD      (UAFP),A
0517  C9         1756          RET
                 1757  ;
                 1758  DECODE:
                 1759  ; Decode bit 7¢4 of A register.
                 1760  ; Each bit is extented to 4 bits.
                 1761  ; 0 becomes 0000, 1 becomes 0001.
                 1762  ; The output is stored in HL, which
                 1763  ; is 16 bits in length.  Also, after
                 1764  ; execution, bit 7¢4 of A register are
                 1765  ; bit 3¢0 of A before execution.
                 1766  ; Register AF,B,HL are destroyed.
                 1767
0518  0604       1768          LD      B,4     ;Loop 4 times.
051A  29         1769  DRL4    ADD     HL,HL   ;Clear rightmost 3
                 1770                          ;bits of HL.
051B  29         1771          ADD     HL,HL
051C  29         1772          ADD     HL,HL
051D  07         1773          RLCA
051E  ED6A       1774          ADC     HL,HL   ;The 4th bit of HL
                 1775                          ;is determined by carry
                 1776                          ;flag, which is the MSB
                 1777                          ;of A register.
0520  10F8       1778          DJNZ    DRL4
0522  C9         1779          RET
                 1780  ;
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 1781 ENCODE:
                 1782 ; Encode HL register.  Each 4 bits of HL
                 1783 ; are encoded to 1 bit. 0000 become 0,
                 1784 ; 0001 become 1.  The result is stored
                 1785 ; in bit 3¢0 of A register.  Also, after
                 1786 ; execution, bit 7¢4 of A are bit 3¢0
                 1787 ; before execution.
                 1788 ; Registers AF,B,HL are destroyed.
                 1789
0523  0604       1790         LD      B,4      ;Loop 4 times.
0525  29         1791 ERL4    ADD     HL,HL    ;Shift HL left 4 bits.
                 1792                          ;Bit 12 of HL will be
                 1793                          ;shifted into carry flag.
0526  29         1794         ADD     HL,HL
0527  29         1795         ADD     HL,HL
0528  29         1796         ADD     HL,HL
0529  17         1797         RLA              ;Rotate carry flag into
                 1798                          ;A register.
052A  10F9       1799         DJNZ    ERL4
052C  C9         1800         RET
                 1801 ;
                 1802 ;*****************************************************
                 1803 SUM1:
                 1804 ; Calculate the sum of the data in a memory
                 1805 ; block.  The starting and ending address
                 1806 ; of this block are stored in STEPBF+2 ¢ STEPBF+4.
                 1807 ;   Registers AF,BC,DE,HL are destroyed.
                 1808
052D  CD3A05     1809         CALL    GETPTR   ;Get parameters from
                 1810                          ;step buffer.
0530  D8         1811         RET     C        ;Return if the parameters
                 1812                          ;are illegal.
                 1813 SUM:
                 1814 ; Calculate the sum of a memory block.
                 1815 ; HL contains the starting address of
                 1816 ; this block, BC contains the length.
                 1817 ; The result is stored in A.  Registers
                 1818 ; AF,BC,HL are destroyed.
                 1819
0531  AF         1820         XOR     A        ;Clear A.
0532  86         1821 SUMCAL  ADD     A,(HL)   ;Add
0533  EDA1       1822         CPI
0535  EA3205     1823         JP      PE,SUMCAL
0538  B7         1824         OR      A        ;Clear flags.
0539  C9         1825         RET
                 1826 ;
                 1827 GETPTR:
                 1828 ; Get parameters from step buffer.
                 1829 ; Input: (STEPBF+2) and (STEPBF+3) contain
                 1830 ;        starting address.
                 1831 ;        (STEPBF+4) and (STEPBF+5) contain
                 1832 ;        ending address.
                 1833 ; Output: HL register contains the starting
                 1834 ;        address.
                 1835 ;         BC register contains the length.
                 1836 ;         Carry flay  0 -- BC positive
                 1837 ;                     1 -- BC negative
                 1838 ; Destroyed reg.: AF,BC,DE,HL.
                 1839
053A  21B11F     1840         LD      HL,STEPBF+2
053D  5E         1841 GETP    LD      E,(HL)   ;Load starting address
                 1842                          ;into DE.
053E  23         1843         INC     HL
053F  56         1844         LD      D,(HL)
0540  23         1845         INC     HL
0541  4E         1846         LD      C,(HL)
0542  23         1847         INC     HL       ;Load ending address
                 1848                          ;into HL.
0543  66         1849         LD      H,(HL)
0544  69         1850         LD      L,C
0545  B7         1851         OR      A        ;Clear carry flag.
0546  ED52       1852         SBC     HL,DE    ;Find difference.
                 1853                          ;Carry flag is changed here.
0548  4D         1854         LD      C,L
0549  44         1855         LD      B,H
054A  03         1856         INC     BC       ;Now BC contains the
                 1857                          ;length.
054B  EB         1858         EX      DE,HL    ;Now HL contains the
                 1859                          ;starting address.
054C  C9         1860         RET
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 1861  ;
                 1862  TAPEIN:
                 1863  ; Load a memory block from tape.
                 1864  ; Input: HL -- starting address of the block
                 1865  ;        BC -- length of the block
                 1866  ; Output: Carry flag,1 -- reading error
                 1867  ;                    0 -- no error
                 1868  ; Destroyed reg. -- AF,BC,DE,HL,AF',BC',DE',HL'
                 1869
054D  AF         1870          XOR     A        ;Clear carry flag.
                 1871                           ;At beginning, the reading is
                 1872                           ;no error.
054E  08         1873          EX      AF,AF'
054F  CD5A05     1874  TLOOP   CALL    GETBYTE  ;Read 1 byte from tape.
0552  73         1875          LD      (HL),E   ;Store it into memory.
0553  EDA1       1876          CPI
0555  EA4F05     1877          JP      PE,TLOOP ;Loop until length
                 1878                           ;is zero.
0558  08         1879          EX      AF,AF'
0559  C9         1880          RET
                 1881  ;
                 1882  GETBYTE:
                 1883  ; Read one byte from tape.
                 1884  ; Output: E -- data read
                 1885  ;         Carry of F',1 -- reading error
                 1886  ;                     0 -- no error
                 1887  ; Destroy reg. -- AF,DE,AF',BC',DE',HL'
                 1888  ; Byte format:
                 1889
                 1890  ; start bit bit bit bit bit bit bit bit stop
                 1891  ;  bit   0   1   2   3   4   5   6   7   bit
                 1892
055A  CD6B05     1893          CALL    GETBIT   ;Get start bit.
055D  1608       1894          LD      D,8      ;Loop 8 times.
055F  CD6B05     1895  BLOOP   CALL    GETBIT   ;Get one data bit.
                 1896                           ;Result in carry flag.
0562  CB1B       1897          RR      E        ;Rotate it into E.
0564  15         1898          DEC     D
0565  20F8       1899          JR      NZ,BLOOP
0567  CD6B05     1900          CALL    GETBIT   ;Get stop bit.
056A  C9         1901          RET
                 1902  ;
                 1903  ;
                 1904  GETBIT:
                 1905  ; Read one bit from tape.
                 1906  ; Output: Carry of F,0 -- this bit is 0
                 1907  ;                    1 -- this bit is 1
                 1908  ;         Carry of F',1 -- reading error
                 1909  ;                    0 -- no error
                 1910  ; Destroyed reg. -- AF,AF',BC',DE',HL'
                 1911  ; Bit format:
                 1912
                 1913  ;   0 -- 2K Hz 8 cycles + 1K Hz 2 cycles.
                 1914  ;   1 -- 2K Hz 4 cycles + 1K Hz 4 cycles.
                 1915
056B  D9         1916          EXX     ;Save HL,BC,DE registers
                 1917
                 1918  ; The tape-bit format of both 0 and 1 are
                 1919  ; of the same form:  high freq part + low freq part.
                 1920  ; The difference between 0 and 1 is the
                 1921  ; number high freq cycles and low freq
                 1922  ; cycles.  Thus, a high freq period may has
                 1923  ; two meanings:
                 1924  ;  i) It is used to count the number of high
                 1925  ;     freq cycles of the current tape-bit;
                 1926  ; ii) If a high freq period is detected
                 1927  ;     immediately after a low freq period, then
                 1928  ;     this period is the first cycle of next
                 1929  ;     tape-bit and is used as a terminator of the
                 1930  ;     last tape-bit.
                 1931
                 1932  ; Bit 0 of H register is used to indicate the usage
                 1933  ; of a high freq period.  If this bit is zero, high
                 1934  ; freq period causes counter increment for the current
                 1935  ; tape-bit.  If the high freq part has passed, bit 0
                 1936  ; of H is set and the next high freq period will be used
                 1937  ; as a terminator.
                 1938  ; L register is used to up/down count the number of periods.
                 1939  ; when a high freq period is read, L is increased by
                 1940  ; 1; when a low freq period is read, L is decreased
                 1941  ; by 2. (The time duration for each count is 0.5 ms.)
                 1942  ; At the end of a tape-bit, positive and negative L
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 1943  ; stand for 0 and 1 respectively.
                 1944
056C  210000     1945          LD      HL,0     ;Clear bit 0 of H,
                 1946                           ;Set L to 0.
056F  CD8C05     1947  COUNT   CALL    PERIOD ;Read one period.
0572  14         1948          INC     D        ;The next 2 instructions
                 1949                           ;check if D is zero.  Carry
                 1950                           ;flag is not affected.
0573  15         1951          DEC     D
0574  2011       1952          JR      NZ,TERR ;If D is not zero, jump
                 1953                           ;to error routine TERR.
                 1954                           ;(Because the period is too
                 1955                           ;much longer than that of 1K Hz.)
0576  3806       1956          JR      C,SHORTP ;If the period is short
                 1957                           ;(2K Hz), jump to SHORTP.
0578  2D         1958          DEC     L        ;The period is 1K Hz,
                 1959                           ;decrease L by 2.  And set
                 1960                           ;bit 0 of H to indicate this
                 1961                           ;tape-bit has passed high freq
                 1962                           ;part and reaches its low freq part.
0579  2D         1963          DEC     L
057A  CBC4       1964          SET     0,H
057C  18F1       1965          JR      COUNT
057E  2C         1966  SHORTP  INC     L        ;The period is 2 K Hz,
                 1967                           ;increase L by 1.
057F  CB44       1968          BIT     0,H      ;If the tape-bit has passed
                 1969                           ;its high freq part, high frquency
                 1970                           ;means this bit is all over and
                 1971                           ;next bit has started.
0581  28EC       1972          JR      Z,COUNT
                 1973                           ;L = (# of 2K period) - 2*(# of 1K period)
0583  CB15       1974          RL      L
                 1975                           ; 0 --- NCarry (L positive)
                 1976                           ; 1 ---  Carry (L negative)
                 1977                           ;The positive or negative sign of
                 1978                           ;L corresponds to the tape-bit data.
                 1979                           ;'RL  L' will shift the sign bit of
                 1980                           ;L into carry flag.  After this
                 1981                           ;instruction, the carry flag
                 1982                           ;contains the tape-bit.
0585  D9         1983          EXX     ;Restore BC',DE',HL'
0586  C9         1984          RET
0587  08         1985  TERR    EX      AF,AF'
0588  37         1986          SCF     ;Set carry flag of F' to indicate error.
0589  08         1987          EX      AF,AF'
058A  D9         1988          EXX
058B  C9         1989          RET
                 1990  ;
                 1991  PERIOD:
                 1992  ; Wait the tape to pass one period.
                 1993  ; The time duration is stored in DE.  The
                 1994  ; unit is loop count.  Typical value for
                 1995  ; 2K Hz is 28, for 1K Hz is 56.
                 1996  ; Use (56+28)/2 as threshold.  The returned
                 1997  ; result is in carry flag. (1K -- NC, 2K -- C)
                 1998  ; Register AF and DE are destroyed.
                 1999
058C  110000     2000          LD      DE,0
058F  DB00       2001  LOOPH   IN      A,(KIN) ;Bit 7 of port A is Tapein.
0591  13         2002          INC     DE
0592  17         2003          RLA
0593  38FA       2004          JR      C,LOOPH ;Loop until input goes low.
0595  3EFF       2005          LD      A,11111111B ;Echo the tape input to
                 2006                          ;speaker on  MPF-I.
0597  D302       2007          OUT     (DIGIT),A
0599  DB00       2008  LOOPL   IN      A,(KIN)
059B  13         2009          INC     DE
059C  17         2010          RLA
059D  30FA       2011          JR      NC,LOOPL ;Loop until input goes high.
059F  3E7F       2012          LD      A,01111111B ;Echo the tape input to
                 2013                          ;speaker on MPF-I.
05A1  D302       2014          OUT     (DIGIT),A
05A3  7B         2015          LD      A,E      ;Compare the result with
                 2016                           ;the threshold.
05A4  FE2A       2017          CP      MPERIOD
05A6  C9         2018          RET
                 2019  ;
                 2020  ;*****************************************************
                 2021  TAPEOUT:
                 2022  ; Output a memory block to tape.
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 2104  ; Input: HL -- address to be check.
                 2105  ; Output: Zero flag -- 0, ROM or nonexistant;
                 2106  ;                     1, RAM.
                 2107  ; Destroyed reg.: AF.
                 2108  ; Call: none
                 2109
                 2110  RAMCHK:
05F6  7E         2111          LD      A,(HL)
05F7  2F         2112          CPL
05F8  77         2113          LD      (HL),A
05F9  7E         2114          LD      A,(HL)
05FA  2F         2115          CPL
05FB  77         2116          LD      (HL),A
05FC  BE         2117          CP      (HL)
05FD  C9         2118          RET
                 2119  ;
                 2120  ;*****************************************************
                 2121  ; Function: Scan the keyboard and display.  Loop until
                 2122  ;           a key is detected.  If the some key is already
                 2123  ;           pressed when this routine starts execution,
                 2124  ;           return when next key is entered.
                 2125  ; Input: IX points to the buffer contains display patterns.
                 2126  ;        6 LEDs require 6 byte data.  (IX) contains the
                 2127  ;        pattern for rightmost LED, (IX+5) contains the
                 2128  ;        pattern for leftmost LED.
                 2129  ; Output: internal code of the key pressed.
                 2130  ; Destroyed reg. : AF, B, HL, AF', BC', DE'.
                 2131  ;                  All other registers except IY are also
                 2132  ;                  changed during execution, but they are
                 2133  ;                  restored before return.
                 2134  ; Call: SCAN1
                 2135
                 2136  SCAN:
05FE  DDE5       2137          PUSH    IX      ;Save IX.
0600  21E61F     2138          LD      HL,TEST
0603  CB7E       2139          BIT     7,(HL)  ;This bit is sert if the use
                 2140                          ;has entered illegal key.  The
                 2141                          ;display will be disabled as
                 2142                          ;a warning to the user.  This
                 2143                          ;is done by replacing the display
                 2144                          ;buffer pointer IX by BLANK.
0605  2804       2145          JR      Z,SCPRE
0607  DD21A507   2146          LD      IX,BLANK
                 2147
                 2148  ; Wait until all keys are released for 40 ms.
                 2149  ; (The execution time of SCAN1 is 10 ms,
                 2150  ; 40 = 10 * 4.)
                 2151
060B  0604       2152  SCPRE   LD      B,4
060D  CD2406     2153  SCNX    CALL    SCAN1
0610  30F9       2154          JR      NC,SCPRE ;If any key is pressed, re-load
                 2155                           ;the debounce counter B by 4.
0612  10F9       2156          DJNZ    SCNX
0614  CBBE       2157          RES     7,(HL)  ;Clear error-flag.
0616  DDE1       2158          POP     IX      ;Restore original IX.
                 2159
                 2160  ; Loop until any key is pressed.
                 2161
0618  CD2406     2162  SCLOOP  CALL    SCAN1
061B  38FB       2163          JR      C,SCLOOP
                 2164
                 2165  ; Convert the key-position-code returned by SCAN1 to
                 2166  ; key-internal-code.  This is done by table-lookup.
                 2167  ; The table used is KEYTAB.
                 2168
061D  217B07     2169  KEYMAP  LD      HL,KEYTAB
0620  85         2170          ADD     A,L
0621  6F         2171          LD      L,A
0622  7E         2172          LD      A,(HL)
0623  C9         2173          RET
                 2174  ;
                 2175  ;*****************************************************
                 2176  ; Function: Scan keyboard and display one cycle.
                 2177  ;           Total execution time is about 10 ms (exactly
                 2178  ;           9.95 ms, 17812 clock states @ 1.79 MHz).
                 2179  ; Input: Same as SCAN.
                 2180  ; Output:  i) no key during one scan
                 2181  ;                  Carry flay -- 1
                 2182  ;         ii) key pressed during one scan
                 2183  ;                  Carry flag -- 0,
                 2184  ;                  A -- position code of the key pressed.
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
                 2185  ;                          If more than one key is pressed, A
                 2186  ;                          contains the largest position-code.
                 2187  ;                          (This key is the last key scanned.)
                 2188  ; Destroyed reg: AF, AF', BC', DE'. (see comments on SCAN)
                 2189  ; Call: none.
                 2190
                 2191  SCAN1:
                 2192  ;In hardware, the display and keyboard are
                 2193  ;arranged as a 6 by 6 matrix.  Each cloumn
                 2194  ;corresponds to one LED and six key buttons.
                 2195  ;In normal operation, at most one column is
                 2196  ;active.  The pattern of the active LED is the
                 2197  ;data output on port C of 8255 I.  The data input
                 2198  ;from bit 0C5 of port A are the status of key
                 2199  ;buttons in the active column.  All signals on
                 2200  ;I/O port are active low.
                 2201
0624   37        2202          SCF              ;Set carry flag.
0625   08        2203          EX       AF,AF'
0626   D9        2204          EXX
                 2205
                 2206  ;Carry flag of F' is used to return the status of
                 2207  ;the keyboard.  If any key is pressed during one
                 2208  ;scan, the flag is reset; otherwise, it is set.
                 2209  ;Initially, this flag is set.  A' register is used
                 2210  ;to store the position-code of the key pressed.
                 2211  ;In this routine, 36 key positions are checked one
                 2212  ;by one.  C register contains the code of the key
                 2213  ;being checked.  The value of C is 0 at the beginning,
                 2214  ;and is increased by 1 after each check.  So the code
                 2215  ;ranges from 0 to 23H (total 36 positions).  On each
                 2216  ;check, if the input bit is 0 (key pressed), C register
                 2217  ;is copied into A'.  The carry flag of F' is set also.
                 2218  ;When some key is detected, the key positions after
                 2219  ;this key will still be checked.  So if more than
                 2220  ;one key are pressed during one scan, the code of the
                 2221  ;last one will be returned.
                 2222
0627   0E00      2223          LD       C,0      ;Initial position code
0629   1EC1      2224          LD       E,11000001B ;Scan from rightmost digit.
062B   2606      2225          LD       H,6
                 2226                                   ;to the active column.
062D   7B        2227  KCOL    LD       A,E
062E   D302      2228          OUT      (DIGIT),A        ;Activate the column.
0630   DD7E00    2229          LD       A,(IX)
0633   D301      2230          OUT      (SEG7),A
0635   06C9      2231          LD       B,COLDEL
0637   10FE      2232          DJNZ     $        ;Delay 1.5 ms per digit.
0639   AF        2233          XOR      A        ;Deactivate all display segments
063A   D301      2234          OUT      (SEG7),A
063C   7B        2235          LD       A,E
063D   2F        2236          CPL
063E   F6C0      2237          OR       11000000B
0640   D302      2238          OUT      (DIGIT),A
0642   0606      2239          LD       B,6      ;Each column has 6 keys.
0644   DB00      2240          IN       A,(KIN)  ;Now, bit 0C5 of A contain
                 2241                            ;the status of the 6 keys
                 2242                            ;in the active column.
0646   57        2243          LD       D,A      ;Store A into D.
0647   CB1A      2244  KROW    RR       D        ;Rotate D 1 bit right, bit 0
                 2245                            ; of D will be rotated into
                 2246                            ;carry flag.
0649   3802      2247          JR       C,NOKEY  ;Skip next 2 instructions
                 2248                            ;if the key is not pressed.
                 2249                            ;The next 2 instructions
                 2250                            ;store the current position-code
                 2251                            ;into A' and reset carry flag
                 2252                            ;of F' register.
064B   79        2253          LD       A,C      ;Key-in, get key position.
064C   08        2254          EX       AF,AF'   ;Save A & Carry in AF'.
064D   0C        2255  NOKEY   INC      C        ;Increase current key-code by 1.
064E   10F7      2256          DJNZ     KROW     ;Loop until 6 keys in the
                 2257                            ;active colums are all checked.
0650   DD23      2258          INC      IX
0652   7B        2259          LD       A,E
0653   E63F      2260          AND      00111111B
0655   CB07      2261          RLC      A
0657   F6C0      2262          OR       11000000B
0659   5F        2263          LD       E,A
065A   25        2264          DEC      H
065B   20D0      2265          JR       NZ,KCOL
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
065D  11FAFF     2266          LD      DE,-6
0660  DD19       2267          ADD     IX,DE    ;Get original IX.
0662  D9         2268          EXX
0663  08         2269          EX      AF,AF'
0664  C9         2270          RET
                 2271  ;
                 2272  ;*************************************************************
                 2273  ; Function: Convert the 2 byte data stored in DE to
                 2274  ;           7-segament display format.  The output is stored
                 2275  ;           in the address field of DISPBF (display buffer),
                 2276  ;           most significiant digit in DISPBF+5.
                 2277  ;           This routine is usually used by monitor only.
                 2278  ; Destroyed reg: AF, HL.
                 2279  ; Call: HEX7SG
                 2280
                 2281  ADDRDP:
0665  21B81F     2282          LD      HL,DISPBF+2
0668  7B         2283          LD      A,E
0669  CD7806     2284          CALL    HEX7SG
066C  7A         2285          LD      A,D
066D  CD7806     2286          CALL    HEX7SG
0670  C9         2287          RET
                 2288  ;
                 2289  ;*************************************************************
                 2290  ; Function: Convert the data stored in A to 7-segament
                 2291  ;           display format.  1 byte is converted to 2
                 2292  ;           digits.  The result is stored in the data
                 2293  ;           field of display buffer (DISPBF).
                 2294  ;           This routine is usually used by monitor only.
                 2295  ; Destroyed reg: AF, HL.
                 2296  ; Call: HEX7SG
                 2297
                 2298  DATADP:
0671  21B61F     2299          LD      HL,DISPBF
0674  CD7806     2300          CALL    HEX7SG
0677  C9         2301          RET
                 2302  ;
                 2303  ;*************************************************************
                 2304  ; Function: Convert binary data to 7-segament display
                 2305  ;           format.
                 2306  ; Input: 1 byte in A register.
                 2307  ;        HL points to the result buffer.
                 2308  ; Output: Pattern for 2 digits.  Low order digit in (HL),
                 2309  ;         high order digit in (HL+1).
                 2310  ;         HL becomes HL+2.
                 2311  ; Destory reg: AF, HL.
                 2312  ; Call: HEX7
                 2313
                 2314  HEX7SG:
0678  F5         2315          PUSH    AF
0679  CD8906     2316          CALL    HEX7
067C  77         2317          LD      (HL),A
067D  23         2318          INC     HL
067E  F1         2319          POP     AF
067F  0F         2320          RRCA
0680  0F         2321          RRCA
0681  0F         2322          RRCA
0682  0F         2323          RRCA
0683  CD8906     2324          CALL    HEX7
0686  77         2325          LD      (HL),A
0687  23         2326          INC     HL
0688  C9         2327          RET
                 2328  ;
                 2329  ;*************************************************************
                 2330  ; Function: Convert binary data to 7-segament display
                 2331  ;           format.
                 2332  ; Input: A -- LSB 4 bits contains the binary data
                 2333  ; Output: A -- display pattern for 1 digit.
                 2334  ; Destroyed reg: AF
                 2335  ; Call: none
                 2336
                 2337  HEX7:
0689  E5         2338          PUSH    HL
068A  21F007     2339          LD      HL,SEGTAB
068D  E60F       2340          AND     0FH
068F  85         2341          ADD     A,L
0690  6F         2342          LD      L,A
0691  7E         2343          LD      A,(HL)
0692  E1         2344          POP     HL
0693  C9         2345          RET
                 2346  ;
                 2347  ;
```

```
LOC    OBJ CODE M STMT SOURCE STATEMENT
                2348  ;*********************************************************
                2349  ; Function: RAM 1800-1FFF self-check.
                2350  ; Input: none
                2351  ; Output: none
                2352  ; Destroyed reg: AF, BC, HL
                2353  ; Call: RAMCHK
                2354
                2355  RAMTEST:
0694   210018   2356          LD      HL,1800H
0697   010008   2357          LD      BC,800H
069A   CDF605   2358  RAMT    CALL    RAMCHK
069D   2801     2359          JR      Z,TNEXT
069F   76       2360          HALT            ;If error.
06A0   EDA1     2361  TNEXT   CPI
06A2   EA9A06   2362          JP      PE,RAMT
06A5   C7       2363          RST     0       ;Display 'uPF--1'.
                2364  ;
                2365  ;*********************************************************
                2366  ;Monitor ROM self-check.  Add the data of address
                2367  ;0000 ¢ 0800.  If the sum equals to 0.  Reset the monitor
                2368  ;and display 'uPF--1'.  If the sum is not 0, which
                2369  ;indicates error, HALT.
                2370  ;Input: none.
                2371  ;Output: none.
                2372  ;Destroyed registers: AF, BC, HL.
                2373  ;Call: SUM.
                2374
                2375  ROMTEST:
06A6   210000   2376          LD      HL,0
06A9   010008   2377          LD      BC,800H
06AC   CD3105   2378          CALL    SUM
06AF   2801     2379          JR      Z,SUMOK
06B1   76       2380          HALT            ;If error.
06B2   C7       2381  SUMOK   RST     0       ;Display 'uPF--1'.
06B3   32E51F   2382  INI3    LD      (POWERUP);A ;Load power-code into
                2383                              ;(POWERUP).  The monitor
                2384                              ;uses the location to decide
                2385                              ;whether a reset signal is
                2386                              ;on power-up.
06B6   3E55     2387          LD      A,55H
06B8   32F01F   2388          LD      (BEEPSET),A
06BB   3E44     2389          LD      A,44H
06BD   32F11F   2390          LD      (FBEEP),A   ;Beep frequency when key is
                2391                              ;pressed.
06C0   21F21F   2392          LD      HL,TBEEP
06C3   362F     2393          LD      (HL),2FH    ;Time duration of beep when
06C5   23       2394          INC     HL
06C6   3600     2395          LD      (HL),0
                2396                              ;key is pressed.
06C8   C3D803   2397          JP      INI4
                2398
06CB   F5       2399  BEEP    PUSH    AF
06CC   21F11F   2400          LD      HL,FBEEP
06CF   4E       2401          LD      C,(HL)
06D0   2AF21F   2402          LD      HL,(TBEEP)
06D3   3AF01F   2403          LD      A,(BEEPSET)
06D6   FE55     2404          CP      55H
06D8   2003     2405          JR      NZ,NOTONE   ;There is no beep sound when
                2406                              ;the key is pressed if data
                2407                              ;of (BEEPSET) is not 55H
06DA   CDE405   2408          CALL    TONE
                2409  NOTONE:
06DD   F1       2410          POP     AF
06DE   C3E900   2411          JP      KEYEXEC ;After a key is detected,determine
                2412                          ;what action should the monitor take.
                2413                          ;KEYEXEC uses the next 3 factors
                2414                          ;to get the entry point of proper
                2415                          ;service routine :key-code, STATE
                2416                          ;and STMINOR (Minor-State).
                2417  ; Below are the branch tables for each key and
                2418  ; state.  The first entry of each table is
                2419  ; a base address, other entrys are the offset to
                2420  ; this address.  Offset is only one byte long,
                2421  ; which is much shorter than the 2-byte address.
                2422  ; This can save the monitor code space.
                2423
0737            2424  KSUBFUN ORG     0737H
0737   1B01     2425          DEFW    KINC
0739   00       2426          DEFB    -KINC+KINC
073A   05       2427          DEFB    -KINC+KDEC
073B   0A       2428          DEFB    -KINC+KGO
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
073C  DF         2429          DEFB    -KINC+KSTEP
073D  1A         2430          DEFB    -KINC+KDATA
073E  2C         2431          DEFB    -KINC+KSBR
073F  42         2432          DEFB    -KINC+KINS
0740  7B         2433          DEFB    -KINC+KDEL
0741  C201       2434  KFDN    DEFW    KPC
0743  00         2435          DEFB    -KPC+KPC
0744  1C         2436          DEFB    -KPC+KADDR
0745  0A         2437          DEFB    -KPC+KCBR
0746  14         2438          DEFB    -KPC+KREG
0747  20         2439          DEFB    -KPC+KMV
0748  20         2440          DEFB    -KPC+KRL.
0749  26         2441          DEFB    -KPC+KWT
074A  26         2442          DEFB    -KPC+KRT
074B  EC01       2443  HTAB    DEFW    HFIX
074D  00         2444          DEFB    -HFIX+HFIX
074E  16         2445          DEFB    -HFIX+HAD
074F  03         2446          DEFB    -HFIX+HDA
0750  26         2447          DEFB    -HFIX+HRGFIX
0751  34         2448          DEFB    -HFIX+HMV
0752  34         2449          DEFB    -HFIX+HRL
0753  34         2450          DEFB    -HFIX+HWT
0754  34         2451          DEFB    -HFIX+HRT
0755  26         2452          DEFB    -HFIX+HRGAD
0756  44         2453          DEFB    -HFIX+HRGDA
0757  3D02       2454  ITAB    DEFW    IFIX
0759  00         2455          DEFB    -IFIX+IFIX
075A  03         2456          DEFB    -IFIX+IAD
075B  03         2457          DEFB    -IFIX+IDA
075C  00         2458          DEFB    -IFIX+IRGFIX
075D  0E         2459          DEFB    -IFIX+IMV
075E  0E         2460          DEFB    -IFIX+IRL
075F  0E         2461          DEFB    -IFIX+IWT
0760  0E         2462          DEFB    -IFIX+IRT
0761  1F         2463          DEFB    -IFIX+IRGAD
0762  1F         2464          DEFB    -IFIX+IRGDA
0763  6B02       2465  DTAB    DEFW    DFIX
0765  00         2466          DEFB    -DFIX+DFIX
0766  03         2467          DEFB    -DFIX+DAD
0767  03         2468          DEFB    -DFIX+DDA
0768  00         2469          DEFB    -DFIX+DRGFIX
0769  0E         2470          DEFB    -DFIX+DMV
076A  0E         2471          DEFB    -DFIX+DRL
076B  0E         2472          DEFB    -DFIX+DWT
076C  0E         2473          DEFB    -DFIX+DRT
076D  1F         2474          DEFB    -DFIX+DRGAD
076E  1F         2475          DEFB    -DFIX+DRGDA
076F  9902       2476  GTAB    DEFW    GFIX
0771  00         2477          DEFB    -GFIX+GFIX
0772  03         2478          DEFB    -GFIX+GAD
0773  03         2479          DEFB    -GFIX+GDA
0774  00         2480          DEFB    -GFIX+GRGFIX
0775  4B         2481          DEFB    -GFIX+GMV
0776  6D         2482          DEFB    -GFIX+GRL
0777  8B         2483          DEFB    -GFIX+GWT
0778  C1         2484          DEFB    -GFIX+GRT
0779  00         2485          DEFB    -GFIX+GRGAD
077A  00         2486          DEFB    -GFIX+GRGDA
                 2487
                 2488  ; Key-position-code to key-internal-code conversion table.
                 2489
                 2490  KEYTAB:
077B  03         2491  K0      DEFB    03H     ;HEX_3
077C  07         2492  K1      DEFB    07H     ;HEX_7
077D  0B         2493  K2      DEFB    0BH     ;HEX_B
077E  0F         2494  K3      DEFB    0FH     ;HEX_F
077F  20         2495  K4      DEFB    20H     ;NOT USED
0780  21         2496  K5      DEFB    21H     ;NOT USED
0781  02         2497  K6      DEFB    02H     ;HEX_2
0782  06         2498  K7      DEFB    06H     ;HEX_6
0783  0A         2499  K8      DEFB    0AH     ;HEX_A
0784  0E         2500  K9      DEFB    0EH     ;HEX_E
0785  22         2501  K0A     DEFB    22H     ;NOT USED
0786  23         2502  K0B     DEFB    23H     ;NOT USED
0787  01         2503  K0C     DEFB    01H     ;HEX_1
0788  05         2504  K0D     DEFB    05H     ;HEX_5
0789  09         2505  K0E     DEFB    09H     ;HEX_9
078A  0D         2506  K0F     DEFB    0DH     ;HEX_D
078B  13         2507  K10     DEFB    13H     ;STEP
078C  1F         2508  K11     DEFB    1FH     ;TAPERD
078D  00         2509  K12     DEFB    00H     ;HEX_0
078E  04         2510  K13     DEFB    04H     ;HEX_4
```

```
LOC   OBJ CODE M STMT SOURCE STATEMENT
078F  08         2511  K14      DEFB    08H     ;HEX_8
0790  0C         2512  K15      DEFB    0CH     ;HEX_C
0791  12         2513  K16      DEFB    12H     ;GO
0792  1E         2514  K17      DEFB    1EH     ;TAPEWR
0793  1A         2515  K18      DEFB    1AH     ;CBR
0794  18         2516  K19      DEFB    18H     ;PC
0795  1B         2517  K1A      DEFB    1BH     ;REG
0796  19         2518  K1B      DEFB    19H     ;ADDR
0797  17         2519  K1C      DEFB    17H     ;DEL
0798  1D         2520  K1D      DEFB    1DH     ;RELA
0799  15         2521  K1E      DEFB    15H     ;SBR
079A  11         2522  K1F      DEFB    11H     ;-
079B  14         2523  K20      DEFB    14H     ;DATA
079C  10         2524  K21      DEFB    10H     ;+
079D  16         2525  K22      DEFB    16H     ;INS
079E  1C         2526  K23      DEFB    1CH     ;MOVE
                 2527  ;
                 2528  ;
                 2529  ;
                 2530  ;
079F  30         2531  MPF_I    DEFB    030H    ;'1'
07A0  02         2532           DEFB    002H    ;'-'
07A1  02         2533           DEFB    002H    ;'-'
07A2  0F         2534           DEFB    0FH     ;'F'
07A3  1F         2535           DEFB    1FH     ;'P'
07A4  A1         2536           DEFB    0A1H    ;'u'
07A5  00         2537  BLANK    DEFB    0
07A6  00         2538           DEFB    0
07A7  00         2539           DEFB    0
07A8  00         2540           DEFB    0
07A9  00         2541  ERR_     DEFB    0
07AA  00         2542           DEFB    0
07AB  03         2543           DEFB    3       ;'R'
07AC  03         2544           DEFB    3       ;'R'
07AD  8F         2545           DEFB    8FH     ;'E'
07AE  02         2546           DEFB    2       ;'-'
07AF  1F         2547  SYS_SP   DEFB    1FH     ;'P'
07B0  AE         2548           DEFB    0AEH    ;'S'
07B1  02         2549           DEFB    02H     ;'-'
07B2  AE         2550           DEFB    0AEH    ;'S'
07B3  B6         2551           DEFB    0B6H    ;'Y'
07B4  AE         2552           DEFB    0AEH    ;'S'
07B5  1F         2553  ERR_SP   DEFB    1FH     ;'P'
07B6  AE         2554           DEFB    0AEH    ;'S'
07B7  02         2555           DEFB    02      ;'-'
07B8  03         2556           DEFB    03      ;'R'
07B9  03         2557           DEFB    03      ;'R'
07BA  8F         2558           DEFB    8FH     ;'E'
07BB  00         2559           DEFB    0
07BC  AE         2560  STEPTAB  DEFB    0AEH    ;'S'
07BD  8F         2561           DEFB    08FH    ;'E'
07BE  B3         2562           DEFB    0B3H    ;'D'
07BF  00         2563           DEFB    0
07C0  AE         2564           DEFB    0AEH    ;'S'
07C1  B3         2565           DEFB    0B3H    ;'D'
07C2  00         2566           DEFB    0
07C3  00         2567           DEFB    0
07C4  0F         2568           DEFB    0FH     ;'F'
07C5  AE         2569           DEFB    0AEH    ;'S'
07C6  8F         2570           DEFB    08FH    ;'E'
07C7  00         2571           DEFB    0
07C8  0F         2572           DEFB    0FH     ;'F'
07C9  00         2573           DEFB    0
07CA  00         2574  REG_     DEFB    0
07CB  00         2575           DEFB    0
07CC  02         2576           DEFB    02H     ;'-'
07CD  BE         2577           DEFB    0BEH    ;'G'
07CE  8F         2578           DEFB    08FH    ;'E'
07CF  03         2579           DEFB    03H     ;'R'
07D0  0F3F       2580  RGTAB    DEFW    3F0FH   ;'AF'
07D2  8DA7       2581           DEFW    0A78DH  ;'BC'
07D4  8FB3       2582           DEFW    0B38FH  ;'DE'
07D6  8537       2583           DEFW    3785H   ;'HL'
07D8  4F3F       2584           DEFW    3F4FH   ;'AF.'
07DA  CDA7       2585           DEFW    0A7CDH  ;'BC.'
07DC  CFB3       2586           DEFW    0B3CFH  ;'DE.'
07DE  C537       2587           DEFW    37C5H   ;'HL.'
07E0  0730       2588           DEFW    3007H   ;'IX'
07E2  B630       2589           DEFW    30B6H   ;'IY'
07E4  1FAE       2590           DEFW    0AE1FH  ;'SP'
07E6  0F30       2591           DEFW    300FH   ;'IF'
07E8  370F       2592           DEFW    0F37H   ;'FH'
```

```
 LOC   OBJ CODE M STMT SOURCE STATEMENT
07EA   850F       2593          DEFW    0F85H    ;'FL'
07EC   770F       2594          DEFW    0F77H    ;'FH.'
07EE   C50F       2595          DEFW    0FC5H    ;'FL.'
07F0   BD         2596  SEGTAB  DEFB    0BDH     ;'0'
07F1   30         2597          DEFB    30H      ;'1'
07F2   9B         2598          DEFB    09BH     ;'2'
07F3   BA         2599          DEFB    0BAH     ;'3'
07F4   36         2600          DEFB    36H      ;'4'
07F5   AE         2601          DEFB    0AEH     ;'5'
07F6   AF         2602          DEFB    0AFH     ;'6'
07F7   38         2603          DEFB    38H      ;'7'
07F8   BF         2604          DEFB    0BFH     ;'8'
07F9   BE         2605          DEFB    0BEH     ;'9'
07FA   3F         2606          DEFB    3FH      ;'A'
07FB   A7         2607          DEFB    0A7H     ;'B'
07FC   8D         2608          DEFB    08DH     ;'C'
07FD   B3         2609          DEFB    0B3H     ;'D'
07FE   8F         2610          DEFB    08FH     ;'E'
07FF   0F         2611          DEFB    0FH      ;'F'
                  2612  ;
                  2613  ;*****************************************************
                  2614  ;SYSTEM RAM AREA:
1F9F              2615  USERSTK ORG     1F9FH
1F9F              2616          DEFS    16
1FAF              2617  SYSSTK: ORG     1FAFH
1FAF              2618  STEPBF  DEFS    7
1FB6              2619  DISPBF  DEFS    6
                  2620  REGBF:
1FBC              2621  USERAF  DEFS    2
1FBE              2622  USERBC  DEFS    2
1FC0              2623  USERDE  DEFS    2
1FC2              2624  USERHL  DEFS    2
1FC4              2625  UAFP    DEFS    2
1FC6              2626  UBCP    DEFS    2
1FC8              2627  UDEP    DEFS    2
1FCA              2628  UHLP    DEFS    2
1FCC              2629  USERIX  DEFS    2
1FCE              2630  USERIY  DEFS    2
1FD0              2631  USERSP  DEFS    2
1FD2              2632  USERIF  DEFS    2
1FD4              2633  FLAGH   DEFS    2
1FD6              2634  FLAGL   DEFS    2
1FD8              2635  FLAGHP  DEFS    2
1FDA              2636  FLAGLP  DEFS    2
1FDC              2637  USERPC  DEFS    2
                  2638  ;
1FDE              2639  ADSAVE  DEFS    2        ;Contains the address being
                  2640                           ;displayed now.
1FE0              2641  BRAD    DEFS    2        ;Break point address
1FE2              2642  BRDA    DEFS    1        ;Data of break point address
1FE3              2643  STMINOR DEFS    1        ;Minor state
1FE4              2644  STATE   DEFS    1        ;State
1FE5              2645  POWERUP DEFS    1        ;Power-up initialization
1FE6              2646  TEST    DEFS    1        ;Flag, bit 0 -- set when function
                  2647                           ;      or subfunction key is hit.
                  2648                           ;      bit 7 -- set when illegal key
                  2649                           ;            is entered.
1FE7              2650  ATEMP   DEFS    1        ;Temporary storage
1FE8              2651  HLTEMP  DEFS    2        ;Temporary storage
1FEA              2652  TEMP    DEFS    4        ;See comments on routine GDA.
1FEE              2653  IM1AD   DEFS    2        ;Contains the address of Opcode 'FF'
                  2654                           ;service routine. (RST 38H, mode
                  2655                           ;1 interrupt, etc.)
1FF0              2656  BEEPSET DEFS    1        ;Default value is 55H
1FF1              2657  FBEEP   DEFS    1        ;Beep frequency
1FF2              2658  TBEEP   DEFS    2        ;Time duration of beep
                  2659          END
```

CROSS REFERENCE

SYMBOL	VAL	M	DEFN	REFS									
ADDRDP	0665		2281	1262	1462	1523	1636						
ADSAVE	1FDE		2639	283	596	618	681	699	732	745	793	822	839
				914	918	969	973	1077	1176	1458	1633		
ATEMP	1FE7		2650	221	238	274	280						
BEEP	06CB		2399	384									
BEEPSE	1FF0		2656	2388	2403								
BITEND	05D9		2067	2061									
BLANK	07A5		2537	1349	2146								
BLOOP	055F		1895	1899									
BR1	0115		514	525	535	545							
BRAD	1FE0		2641	229	364	602	1024	1386	1474				
BRANCH	03B0		1301	431	455	515							
BRDA	1FE2		2642	230	362	1477							
BRRSTO	00D4		362	351									
BRTEST	0421		1470										
CLRBR	03DE		1378	742									
COLDEL	00C9		27	2231									
CONT28	003E		217	163									
COUNT	056F		1947	1965	1972								
DAD	026E		968	2467									
DATADP	0671		2298	1467									
DDA	026E		969	2468									
DECODE	0518		1758	1733	1735	1738	1740						
DFIX	026B		963	2465	2466	2466	2467	2468	2469	2470	2471	2472	2473
				2474	2475								
DIGIT	0002		16	115	237	279	1091	2007	2014	2094	2230	2238	
DISPBF	1FB6		2619	348	441	858	1447	1454	1483	1527	1535	1642	1673
				2282	2299								
DMV	0279		981	2470									
DOMV	0187		672	724									
DRGAD	028A		995	2474									
DRGDA	028A		996	2475									
DRGFIX	026B		964	2469									
DRGNA	0295		1006	1004									
DRL	0279		980	2471									
DRL4	051A		1769	1778									
DRT	0279		978	2473									
DSTEP	0286		992	989									
DTAB	0763		2465	534									
DWT	0279		979	2472									
EIDI	02B2		1048	1046									
ENCODE	0523		1781	1746	1748	1752	1754						
ENDFUN	031C		1175	1135	1141	1219							
ENDTAP	034D		1218	1298									
ERL4	0525		1791	1799									
ERROR	0353		1221	1119	1167	1189	1285	1288	1296				
ERR	07A9		2541	1221									
ERR_SP	07B5		2553	329									
F1KHZ	0041		29	2086									
F2KHZ	001F		31	2089									
FBEEP	1FF1		2657	2390	2400								
FCONV	04C4		1687	756	1655								
FCONV1	04DA		1732										
FCONV2	04F9		1743	1731									
FILEDP	038A		1265	1266									
FLAGH	1FD4		2633	1734	1743								
FLAGHP	1FD8		2635	1739	1751								
FLAGL	1FD6		2634	1736	1747								
FLAGLP	1FDA		2636	1741	1753								
FLAGX	04D6		1722	1700									
GAD	029C		1023	2478									
GDA	029C		1024	2479									
GETBIT	056B		1904	1893	1895	1900							
GETBYT	055A		1882	1874									
GETP	053D		1841	1113									
GETPTR	053A		1827	1210	1278	1809							
GFIX	0299		1017	2476	2477	2477	2478	2479	2480	2481	2482	2483	2484
				2485	2486								
GMV	02E4		1112	672	2481								
GRGAD	0299		1019	2485									
GRGDA	0299		1020	2486									
GRGFIX	0299		1018	2480									
GRL	0306		1144	2482									
GRT	035A		1226	2484									
GTAB	076F		2476	544									
GWT	0324		1182	2483									
HAD	0202		839	2445									
HDA	01EF		822	2446									
HEX7	0689		2337	2316	2324								
HEX7SG	0678		2314	2284	2286	2300							
HFIX	01EC		816	2443	2444	2444	2445	2446	2447	2448	2449	2450	2451
				2452	2453								

```
CROSS REFERENCE
SYMBOL   VAL M DEFN REFS
HLTEMP 1FE8   2651  162  239  281  287
HMV    0220    871 2448
HRGAD  0212    855 2452
HRGDA  0230    888 2453
HRGFIX 0212    856 2447
HRL    0220    870 2449
HRT    0220    868 2451
HTAB   074B   2443  513
HWT    0220    869 2450
IAD    0240    913 2456
IDA    0240    914 2457
IFIX   023D    908 2454 2455 2455 2456 2457 2458 2459 2460 2461 2462
                   2463 2464
IGNORE 03BB   1336  557  580  594  600  616  630  643  692  708  719
                    816  825  910  936  965  991 1020
IM1AD  1FEE   2653  207 1373
IMV    024B    926 2459
INI    03C1   1347  123
INI1   03C7   1363 1368
INI2   03C9   1364 1365
INI3   06B3   2382 1371
INI4   03D8   1372 2397
IRGAD  025C    941 2463
IRGDA  025C    942 2464
IRGFIX 023D    909 2458
IRGNA  0267    952  950
IRL    024B    925 2460
IRT    024B    923 2462
ISTEP  0258    937  932
ITAB   0757   2454  524
IWT    024B    924 2461
K0     077B   2491
K0A    0785   2501
K0B    0786   2502
K0C    0787   2503
K0D    0788   2504
K0E    0789   2505
K0F    078A   2506
K1     077C   2492
K10    078B   2507
K11    078C   2508
K12    078D   2509
K13    078E   2510
K14    078F   2511
K15    0790   2512
K16    0791   2513
K17    0792   2514
K18    0793   2515
K19    0794   2516
K1A    0795   2517
K1B    0796   2518
K1C    0797   2519
K1D    0798   2520
K1E    0799   2521
K1F    079A   2522
K2     077D   2493
K20    079B   2523
K21    079C   2524
K22    079D   2525
K23    079E   2526
K3     077E   2494
K4     077F   2495
K5     0780   2496
K6     0781   2497
K7     0782   2498
K8     0783   2499
K9     0784   2500
KADDR  01DE    764 2436
KCBR   01CC    738 2437
KCOL   062D   2227 2265
KDATA  0135    565 2430
KDEC   0120    528 2427
KDEL   0196    686 2433
KEYEXE 00E9    392 2411
KEYMAP 061D   2169
KEYTAB 077B   2490 2169
KFUN   0741   2434  453
KGO    0125    538 2428
KHEX   0111    507  403
KIN    0000     18 2001 2008 2240
KINC   011B    518 2425 2426 2426 2427 2428 2429 2430 2431 2432 2433
```

CROSS REFERENCE

SYMBOL	VAL	M	DEFN	REFS									
KINS	015D		610	2432									
KMV	01E2		787	2439									
KPC	01C2		727	2434	2435	2435	2436	2437	2438	2439	2440	2441	2442
KREG	01D6		750	2438									
KRL	01E2		790	2440									
KROW	0647		2244	2256									
KRT	01E8		801	2442									
KSBR	0147		587	2431									
KSTEP	012A		548	2429									
KSUBFU	0737		2424	430									
KWT	01E8		797	2441									
LEAD	0360		1229	1240	1258	1272							
LEAD1	0367		1236	1246									
LEAD2	0371		1250	1251									
LOCPT	04A5		1651	1648									
LOCRG	04BE		1682	1628									
LOCRGB	04BB		1676	888									
LOCSTB	0455		1538	871	1519								
LOCSTN	045F		1550	930	988	1531							
LOOPH	058F		2001	2004									
LOOPL	0599		2008	2011									
MAIN	00DE		379	387									
MEMDP1	0402		1444	768	851								
MEMDP2	040B		1451	371	574	604	682	733	746	835	919	974	1177
MPERIO	002A		33	2017									
MPF_I	079F		2531	258									
MVUP	0300		1136	1126									
NMI	0066		266	174	1372								
NOKEY	064D		2255	2247									
NOTONE	06DD		2409	2405									
OLOOP	05B7		2042	2045									
ONE_1K	0004		47	2066									
ONE_2K	0004		48	2064									
OUT0	05C9		2057										
OUT1	05D2		2063	2056									
OUTBIT	05C4		2050	2041	2043	2047							
OUTBYT	05B1		2033	2028									
P8255	0003		15	107	276								
PERIOD	058C		1991	1236	1250	1947							
POWERU	1FE5		2645	121	2382								
PRECL1	03EE		1402	827	890	1422							
PRECL2	03FA		1416	840	875								
PREOUT	02A3		1036	562									
PREPC	0021		133	131									
PWCODE	00A5		19	122	1370								
RAMCHK	05F6		2110	130	331	334	598	628	707	824	2358		
RAMT	069A		2358	2362									
RAMTES	0694		2355										
REGBF	1FBC		2620	1049	1682								
REGDP8	0473		1600	864									
REGDP9	0477		1606	582	897	952	1006						
RDC	07CA		2574	753									
RESET1	0032		181	140									
RESET2	0054		248	183									
RGNADP	04AE		1659	1624									
RGSAVE	0074		281										
RGSTIN	0479		1611	1604									
RGTAB	07D0		2580	1666									
ROMTES	06A6		2375										
RST28	0028		143										
RST30	0030		156										
RST38	0038		194										
SAV12	0412		1456	1450									
SCAN	05FE		2136	381									
SCAN1	0624		2191	1265	1364	2153	2162						
SCLOOP	0618		2162	2163									
SCNX	060D		2153	2156									
SCPRE	060B		2152	2145	2154								
SEG7	0001		17	1230	1277	2228	2234						
SEGTAB	07F0		2596	2339									
SETIF	00A4		320	318									
SETPT	0434		1487	1491	1530								
SETPT1	0433		1486	1481									
SETSTO	00D0		353	263	332	335	347	1222					
SEORTP	057E		1966	1956									
SKIPH1	0183		645	641									
SKIPH2	01B8		721	717									
SQWAVE	05EA		2094	2099									
STATE	1FE4		2644	361	443	514	1396	1456	1556	1617	1637		
STEPBF	1FAF		2618	623	627	645	680	706	721	723	796	1112	1122
				1132	1144	1153	1192	1198	1218	1227	1254	1260	1292
				1545	1840								

```
CROSS REFERENCE
SYMBOL   VAL M DEFN REFS
STEPDP 043A   1513  805  885  937  992
STEPTA 07BC   2560 1564
STMINO 1FE3   2643  451  859  926  942  981  996 1543 1569 1618 1643
                    1681 1693
SUM    0531   1813 2378
SUM1   052D   1803 1183 1290
SUMCAL 0532   1821 1823
SUMOK  0682   2381 2379
SYSSTK 1FAF   2617  116  322  380
SYS_SP 07AF   2547  341
TAPEIN 054D   1262 1257 1287
TAPEOU 05A7   2021 1203 1213 2030
TBEEP  1FF2   2658 2392 2402
TEMP   1FEA   2652 1036 1048 1079 1094 1228 1267
TERR   0587   1985 1952
TEST   1FE6   2646  252  413 1337 1408 1413 2138
TESTM  03E5   1389  552  569  592  614  690
TESTRG 013E    578  571
TLOOP  054F   1874 1877
TNEXT  06A0   2361 2359
TONE   05E4   2090 2087 2408
TONE1K 05DE   2085 1197 2067
TONE2K 05E2   2088 1209 1217 2059 2065
UAFP   1FC4   2625 1737 1755
UBCP   1FC6   2626
UDEP   1FC8   2627
UHLP   1FCA   2628
USERAF 1FEC   2621 1068 1092 1732 1749
USERBC 1FBE   2622
USERDE 1FC0   2623
USERHL 1FC2   2624
USERIF 1FD2   2632  181  312  320 1037 1069 1717
USERIX 1FCC   2629
USERIY 1FCE   2630  289
USERPC 1FDC   2637  133  285  731
USERSP 1FD0   2631  250  288  328 1067
USERST 1F9F   2615  249  345
ZERO_1 0002     49 2060
ZERO_2 0008     50 2058
ZSUM   0071     20  191
```

3/18/83

E & L
BILL OF MATERIAL LISTING

BOM # 341-0080 THE FOX---MT-80Z

LINE	COMPONENT	DESCRIPTION	QTY	LVL
1	101-0005	BOM REV F		1
2	413-0171	FOX EXPNSN BD ASSY	1	1
3	711-0234	FOX EXPNSN PC BD R/B	1	2
4	503-0098	SN74LS00	1	2
5	503-0121	SN74LS155	1	2
6	503-0124	74LS367	7	2
7	503-0125	RCA CD4042AE	2	2
8	503-0128	SN74LS148	1	2
9	503-0129	SN74LS245	1	2
10	503-0218	SN74LS74	1	2
11	503-0097	SN74LS02	1	2
12	503-0183	74LS373	1	2
13	503-0039	SN7407	1	2
14	503-0120	SN74LS04	1	2
15	- 1			1
16	- 1			1
17	- 1			1
18	511-0039	CCRES 1/4W 1K OHM 5%	3	2
19	511-0062	CCRES 1/4W 10K OHM 5%	7	2
20	511-0080	CCRES 1/4W 100K OHM 5%	2	2
21	516-0002	10K COM SIP 9RES	2	2
22	516-0003	100K COM SIP 9 RES	1	2
23	516-0001	1K SERIES SIP 4RES	4	2
24	- 1			1
25	- 1			1
26	- 1			1
27	- 1			1
28	- 1			1
29	524-0033	10UF TAN CAP 20V 1.5"	4	2
30	520-0008	CER CAP .01 MFD. 50V	3	2
31	- 1			1
32	- 1			1
33	- 1			1
34	- 1			1
35	501-0009	IN4003 DIODE	2	2
36	501-0031	IN995 DIODE	1	2
37	551-0005	MINI RED LED 100 CENT*	16	2
38	633-0093	FOX SK10/IF 44 SKT LBL	1	2
39	633-0094	FOX IF33 SKT LBL	1	2
40	579-0001	8 POSN DIP SWITCH	2	2
41	415-0001	SK10-PL UNIV SKT NEW	1	2
42	415-0018	SK-50 IF-33 HF SKT R/-	1	2
43	415-0021	SK10 HALF/IF44 R/A	1	2
44	540-0020	346-56-520-801 CONNEDA	1	2
45	615-0041	#4X1/4 NYLON SPACER	4	2
46	605-0059	4-40X1/2"FLHD SCREW	7	2
47	605-0075	#4-40X7/8 FL HD	4	2
48	607-0001	#4 SPLIT LW 1/32	11	2
49	606-0009	4-40X3/16" HEXNUT SMAL	11	2

05/18/83

E & L
BILL OF MATERIAL LISTING

BOM # 341-0080 THE FOX---MT-80Z

LINE	COMPONENT	DESCRIPTION	QTY	LVL
50	542-0005	14 PIN DIP SKT STD.	5	2
51	542-0008	16 PIN DIP SKT STD	11	2
52	542-0033	20PIN DIP SKT COM STD	2	2
53	542-0016	40 PIN DIP SKT COM STD	1	2
54	633-0096	FOX SPKR/BD LBL	2	2
55	589-0014	20 COND FLEX CABLE 2LG	2	2
56	571-0015	SPST PWR SWITCH ALCO	1	2
57	572-0010	SPDT PB SW (SHAWDOW)	1	2
58	542-0017	24 PIN DIP SKT COM STD	1	2
59	546-0008	3.5MM PHONE JACK 3COND	1	2
60	- 1			1
61	- 1			1
62	- 1			1
63	412-0016	HOUSING ASSY FOX	1	1
64	620-0012	FOX CASE PFC	1	2
65	633-0092	FOX PACKAGE LABEL	1	2
66	611-0006	RBBR BMPRS SMITH #2451	4	2
67	735-0004	MPROF MOD W/ 9V ADPTR	1	2
68	605-0076	6-19X1/2 PLSTTE BLK	8	2
69	619-0017	FX KYPD RTNR PLTE R/A	1	2
70	619-0018	FINISH FOR 6190017	1	2
71	633-0098	E&L ADDRS & LOGO W/B	1	2
72	- 1			1
73	413-0172	HEAT SINK PLATE ASSY	1	1
74	619-0015	FOX BD HEATSINK	1	2
75	619-0016	FINISH FOR 6190015	1	2
76	605-0025	4-40 X 1/4" PAN HD. SC	3	2
77	606-0009	4-40X3/16" HEXNUT SMAL	3	2
78	504-0013	LM320T-12 7912 CKC	1	2
79	504-0009	LM340T-12/ 7812 CKC	1	2
80	617-0010	TO-220 NYLON BUSHING	2	2
81	616-0012	SIL PAD 7403-09FR-54	2	2
82	504-0005	MOT 7805C	1	2
83	- 1			1
84	- 1			1
85	801-0246	FOX USER MAN R/A	1	1