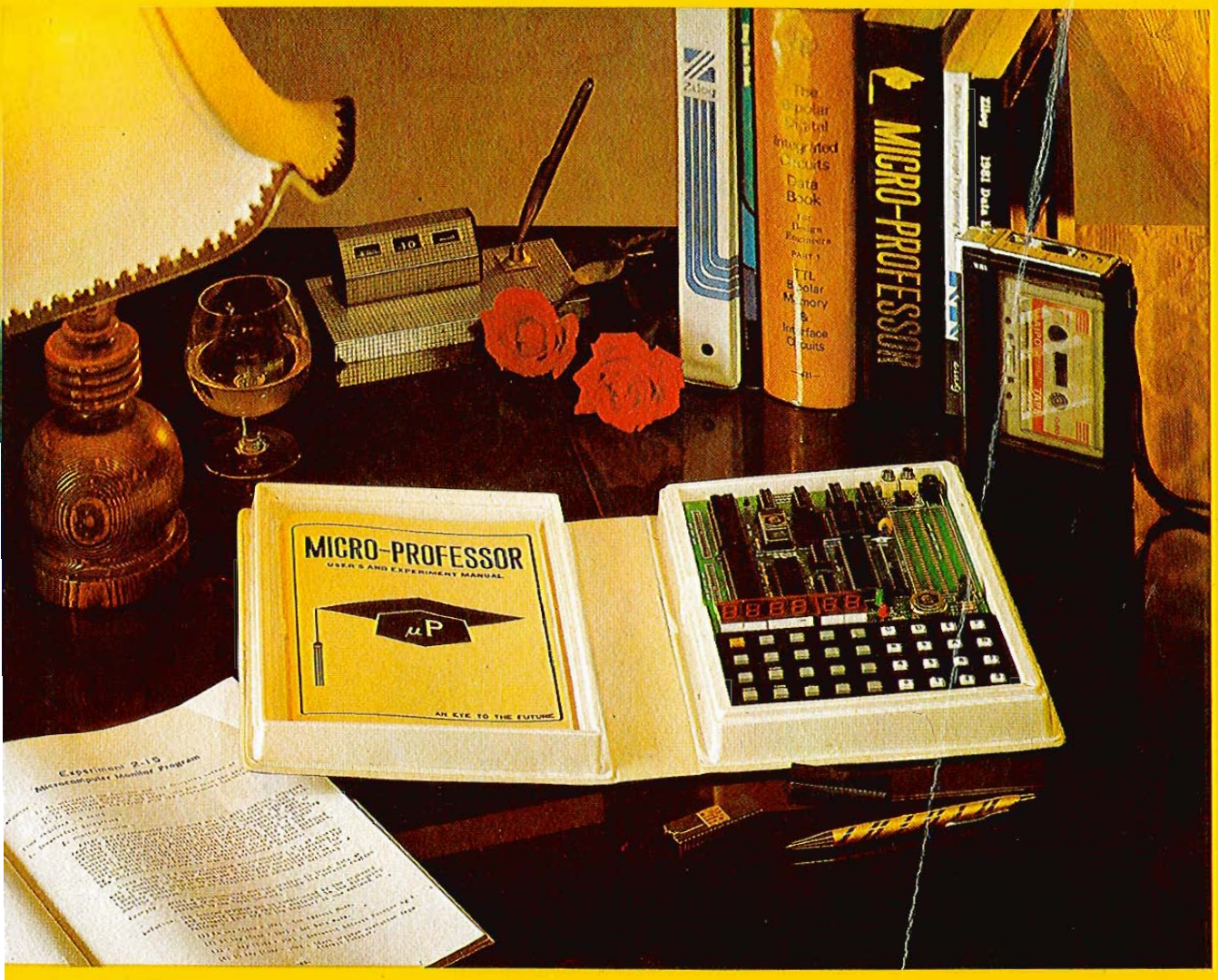
MPF-I

USER'S MANUAL



MPF-I

USER'S MANUAL

THE FIRST 50 YEARS OF THE 20TH CENTURY WITNESSED THE INVENTION OF THE INTERNAL COMBUSTION ENGINE, WHICH GREATLY EXTENDED THE PHYSICAL STRENGTH OF THE HUMAN BODY

IN THE 2ND HALF OF THE CENTURY, THE BIRTH OF THE MICROPORCESSOR FURTHER EXTENDED OUR MENTAL STRENGTH. APPLICATIONS OF THIS AMAZING PRODUCT IN VARIOUS INDUSTRIES HAVE INTRODUCED SO MUCH IMPACT ON OUR LIFE, HENCE, IT IS CALLED THE SECOND INDUSTRIAL REVOLUTION.

CONGRATULATIONS!

Your Micro-Professor will lead you to the world of microprocessor. Unpacking the MPF-I, you will have found the Micro-Professor, an adaptor, and a manual. The standard configuration of your MPF-I includes one MPF-I microcomputer set, two pieces of built-in male header, one unit of book-type package, one AC-DC adaptor, and a copy of User's and Experimental Manual.

In addition to those standard items, three options are for your function expansion which you can buy from local distributor choice:
1) SSB-MPF, which is a speech synthesizer board based on Texas Instruments' TMS5200/5220, and which can reproduce sound and voices stored in its memory.
2) EPB-MPF, which is an EPROM programmer board for TMS2508, TMS2516, TMS2532, Intel 2578, Intel 2716, and Intel 2732.
3) BASIC-MPF, which is a 2K byte tiny BASIC interpreter.

Still, there are some accessories for your choice. You can select
1) SSB-CPK, Z80-CTC (counter and timer) and Z80-PIO (parallel I/O) chip kit.
2) MPF-BBD, 1.42" x 3.15" breadboard.
3) MPF-2KRAM, 2K x 8 RAM 6116, 58725 or others in function equivalent
4) MPF-2KROM, blank 2K bytes EPROM TMS2516, I2716 or equivalent.
5) MPF-4KROM, blank 4K bytes EPROM TMS2532, I2732 or equivalent.

Notes: I. When your MPF-I is in use, the power regulator 7805, which is installed in the upper right corner of the MPF-I, may heat up. A temperature of 70 C is normal. Just keep your hands off the power regulator.

II. Cassettee interface:
1. Use high quality audio tape and tape recorder.
2. When read data from cassette, the volume switch of your tape recorder should be turned to it's maximum.
3. In case you have problems using your cassette recorder for data storage or retrival properly, the battery of cassette recorder may run out of power. Change with new batteries.

If any problem occurs while you use our MPF-I, we wish you to contact us or your local dealers immediately.

NOTE TO USER

This manual is not meant to serve as an introduction to computer programming; the reader is supposed to have had some previous experience on microcomputer and microprocessor. The reader without any previous background on basic concept of computer is suggested to refer "An Introduction to Microcomputers Volume 0 the Beginner's Book" by Adam Osborne, Osborne and Associates Inc. before he starts reading this manual. The reader is also suggested to refer textbook on Z-80 assembly programming such as "Z80 - Assembly Language Programming Manual" published by Zilog Inc.

MANUAL UPDATE

In order to improve the functions and memory capacity of the MPF-I, we have changed the design of the MPF-I slightly. The changes are as follows:

The ROM chip, mounted at board location U6, was upgraded to 2764 whose memory space is from 0000H to 17FFH. Previously, the ROM was 2732 with a address space from 0000H to 0FFFH.

The memory chips which can be installed on board location U7 are of the following types: 2716, 2732, 2764, or 6264.

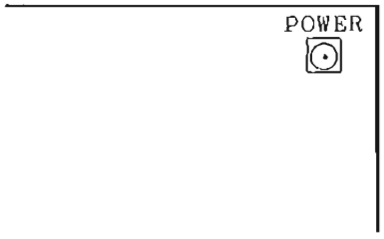
These changes are incorporated in the printed circuit board (PCB) of MPF-I with version number PB820010-9A or higher. The PCB version number is printed in between U6 and the crystal oscillator on the PC board. Please refer to the new schematic diagrams for details.

READ ME FIRST

The manuals that accompany your Micro-Professor are designed for reference and to suggest experiments by showing examples. To get started, it is suggested that you follow the proceedures given below.

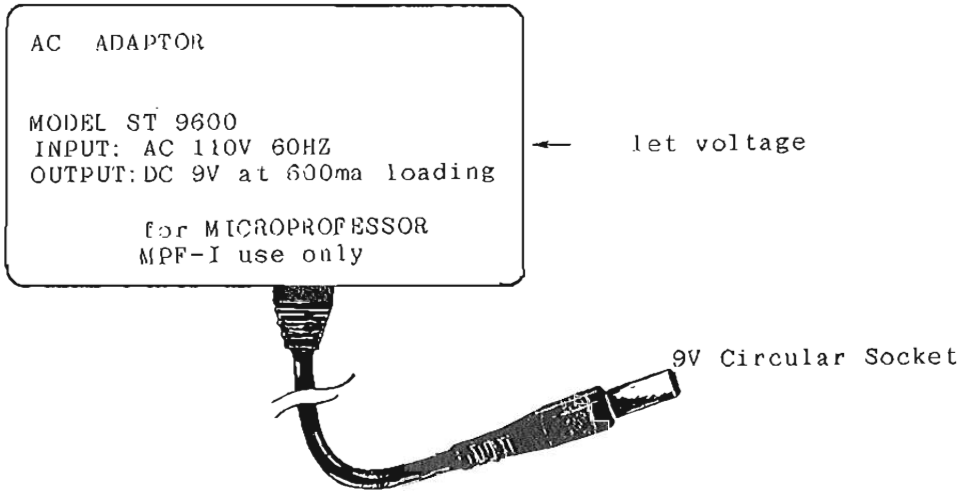
UNPACKING AND INSTALLATION

Open the "book" containing the Micro-Professor(MPF-I). Locate the power connector in the upper right-hand corner Art. A



Art. A Location of the MPF-I power connector

Find the AC adaptor. The adaptor Art. B is a black box labeled "AC ADAPTORH". You should make certain that the voltage input shown on the adaptor matches the voltage supplied by your outlet. In the United States it is assumed (unless a special order is make) that the supply is 117VAC-which is usually referred to as one-ten (110V). You should also check the fre- quency; the label on the adaptor will show the frequency in Hertz(Hz).



Art. B AC ADAPTOR

Plug the 9V circular socket into the power receptacle on the MPF-I. The side opposite the AC Adaptor label is to be plugged into your outlet.

```
**************************************************
* CAUTION DO NOT TOUCH THE PRONGS WHEN PLUGGING  *
* THE AC ADAPTOR INTO YOUR OUTLET                *
**************************************************
```

When power is applied to the MPF-I the following series of patterns should appear

```
                        u       First pattern
                    U   P
                M   P   F
            M   P   F   -
        M   P   F   -   -
    M   P   F   -   -   -       Final pattern
```

Strong background light will make the displays hard to read. If at all possible avoid bright lighting.

TESTING AND FAMILARIZATION

In the exercise below you will be shown how to enter and excute a short program. Performing this exercise will test some of the MPF-I functions and familarize you with the MPF-I. The program used in this section adds two numbers and stores the result in memory.

PROGRAM IN ENGLISH

Load the first number into the A register and the second number into the B register. Add the contents of the B register to the contents of the A register and put the result (sum) in the A register. store the value in the A register in memory location 1830H (H stands for hexadecimal). Finally halt the Micro-Professor.

Source Program in Assembly Language

```
ORG     1800H       ; Start code at 1800 hexadecimal
LD      A,05        ; Load the A register with 5
LD      B,04        ; Load the B register with 4
ADD     A,B         ; A←A + B
LD      (1830H),A   ; Store A at memory location 1830H
HALT                ; Stop execution of program
```

ASSEMBLY LISTING

All program are entered into the MPF-I in hexadecimal. Therefore, you first write your program in assembly language and then translate it into hexadecimal. All of the demonstration programs written in the MPF-I manuals will also list the machine language code - which is in in hexadecimal. A complete assembly listing is shown below.

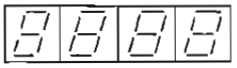
```
LOCATION    MACHINE    STATEMENT    ASSEMBLY
COUNTER     LANGUAGE   NUMBER       LANGUAGE

1800                   1            ORG   1800H        ;Start code at 1800
                                                         hexadecimal
1800        3E05       2            LD    A,05         ;Load the A register
                                                         with 5
1802        0604       3            LD    B,04         ;Load the B register
                                                         with 4
1804        80         4            ADD   A,B          ;A ← A + B
1805        323018     5            LD    (1830H),A    ;Store A at memory
                                                         location 1830H
1808        76         6            HALT               ;Stop execution of
                                                         program
```

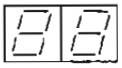
Fig. 0-1 Assembly Language Listing

LOADING THE MACHINE LANGUAGE CODE

You will now enter the machine language code shown in the assembly language listing (Fig. 0-1). If you haven't already done so, connect your MPF-I to the power source. Now press the system reset key [RS]. Section 3.1.1 of the reference manual contains a brief explanation of reset key actions.

Since the available RAM (random access memory) starts at hexadecimal location 1800, the entry of machine language code will start at 1800H. Press the address key [ADDR], a random address will be displayed on the four leftmost digits; these digits will be referred to as


ADDRESS FIELD


DATA FIELD

The address field. Enter the starting address for the machine language code by pressing [1] [8] [0] [0]. The same result can obtained by pressing the program counter key [PC] (this only works when your program starts at 1800H). Now inform the Micro-Professor that data is to be entered by pressing [DATA]. Refer to line 2 of the assembly language listing. Line 2 contains two bytes of machine language code 3E and 05. Key in the first byte by pressing [3] and [E]. The display should now show

[1] [8] [0] [0] [3E]
ADDRESS FIELD DATA FIELD

Advance the address field display by pressing [+]. The display will show

[1] [8] [0] [1] [X] [X] X= unknown data

ADDRESS FIELD DATA FIELD

Enter the second byte of hexadecimal data by pressing [0] then [5]. The display will now show

[1] [8] [0] [1] [0] [5]

ADDRESS FIELD DATA FIELD

Line 3 of the listing also contains two bytes of hexadecimal data; enter these bytes by keying [+], [0], [6] [+], [0], [4] In a similar manor enter the rest of the program, namely

[+] [8] [0] [+] [3] [2] [+] [3] [0] [+] [1] [8] [+] [7] [6]

CHECKING FOR DATA ENTRY ERRORS

The program had been entered. It is wise to check for entry errors. Press [ADDR] [1] [8] [0] [0]. Are the rightmost two displays (dara field) equal to 3E? If not press [DATA] and enter [3] [E]. To examine the next byte press [+]. Is there a 05 in the data field? If the display is correct, continue inspection of all the remaining data using the [+] key. If the present byte or any successive bytes are incorrect, enter the correct data.

Section 3.1.2 contains a formal description of how to enter data.

PROGRAM EXECUTION

There are two ways to begin execution at address 1800H. The simplest is to press [RS], [PC], and then [GO]. The second method allows execution to begin at any address. Press [RS], [ADDR], the beginning execution address, e.g. [1] [8] [0] [0], then [GO]. When you press [GO] the screen will eventually go back less than a second and stay blank. The program has reached the HALT instruction and is waiting for the next operator action.

CHECKING THE RESULTS

To regain control of the keyboard functions press [RS]. The answer to 5+4 was stored at location 1830H. Key in [ADDR] [1] [8] [3] [0]. The dislay should show

[1] [8] [3] [0] [0] [9]
ADDRESS FIELD DATA FIELD

The action of the [PC] and [GO] keys are explained in section 3.1.4 and 3.2.1 respectively.

PROGRAM EXAMPLES

Section 5.10 contains five programming examples. Using the knowledge gained in exercise above enter the hexadecimal code shown in each program and then execute the program. Perform the same steps with the MPF-I Experiment Manual in Experiment-12, 13, 14, 17, 18.

IF YOU MAKE AN ERROR

1) A byte was incorrectly entered. Write the correct over the incorrect byte.

2) One or more bytes were left out. Read section 3.3.3 then remove the bytes one by one.

3) One or more bytes need to be added. Read section 3.3.2 then add each byte.

4) To trace the execution of each instruction see section 3.2.2. Warning: If you are not familiar with the concept of single stepping you will need to read this section several times. You may find it necessary to consult additional learning material.

LEARNING AND EXPERIMENTING

For self learning, proceed to section III. Section III contains a series of experiments. Read the theory (background) of each experiment and then do the exercises. If you do not understand parts of an experiment, do not be discouraged. Some of the experiments are quite advanced. You can refer to the MPF-I Student Workbook.

MPF-I USER'S MANUAL

TABLE OF CONTENTS

1. MPF-I Specification

1.1. Hardware Specifications

(1) CPU: (Central Processing Unit)

Zilog Z-80 CPU with 158 instructions and 2.5 MHz maximum clock rate. The MPF-I system clock is 1.79 MHZ.

(2) ROM: (Read Only Memory)

Single +5V EPROM 2516(2532), total 2K(4K) bytes.
Monitor EPROM Address: 0000-07FF(0FFF).

(3) RAM: (Random Access Memory)

Static RAM: 6116, total 2K bytes.
Basic RAM Address: 1800-1FFF.

(4) Memory Expansion Area:

Single +5V EPROM 2516/2716/2532/2732 EPROM or 6116 static RAM on-Board Expansion Address: 2000-2FFF

(5) I/O Port:

Programmable I/O Port 8255, a total 24 parallel I/O lines are used for keyboard scanning and seven segment LED display control.

I/O addresses: Ø0-Ø3.

Programmable PIO, a total of 16 parallel I/O lines,
I/O address: 80-83H

Programmable CTC, a total of 4 independent counter timers channels, I/O address: 40-43H

(6) Display:

6-digit,Ø.5",7-Segment red LED display

(7) Keyboard:

36 keys including 19 function keys, 16 hex-decimal keys and 1 user-defined key.

(8) Speaker and Speaker Driver Circuits:

A 2.25" - diameter speaker is provided for user's expansion.

(9) User Area:

A 3.5" x 1.36" wire wrapping area is provided for user's expansion.

(10) Audio Tape Interface:

Can be connected to any cassette. Data transmission rate is 165 baud per second (bps).

(11) System Clock Rate:

3.58 MHz crystal is divided by 2, cycle time is 0.56 micro-sec.

(12) System Power Consumption:

Single 5V power supply, current consumption 500 mA.

(13) Main Power Input:

Power adaptor Input 110V 9V/500mA

(14) Physical characteristics

Height	:	1.60 mm (W/O case)
Width	:	15.75 cm (W/O case)
Depth	:	22.30 cm (W/O case)
Weight	:	1.41 lb (With Case)

1.2. Software Specifications

MPF-I contains a high performance 2K-byte monitor program. . It is designed to respond to user input. The monitor commences execution when power is applied. In addition to the key monitor functions, the monitor contains a memory checking routine.

The following is a simple description of the key functions:

(1) [RS] : system reset.

(2) [ADDR] : set memory address.

(3) [REG] : set a register name.

(4) [DATA] : input data to memory or a register.

(5) [PC] : recall program counter.

(6) [+] : increment memory address or a register by one.

(7) [-] : decrement memory address or register by one.

(8) [STEP] : single step the user's program.

(9) [SBR] : set breakpoint in user's program.

(10) [CBR] : clear breakpoint in user's program.

(11) [MON] : terminate the executing program and return the control to the monitor.

(12) [GO] : commence execution at address shown on the display.

(13) [INS] : insert 1 byte into memory.

(14) [DEL] : delete 1 byte from memory.

(15) [MOVE] : move a block of data from one area to another.

(16) [RELA] : relative address calculation.

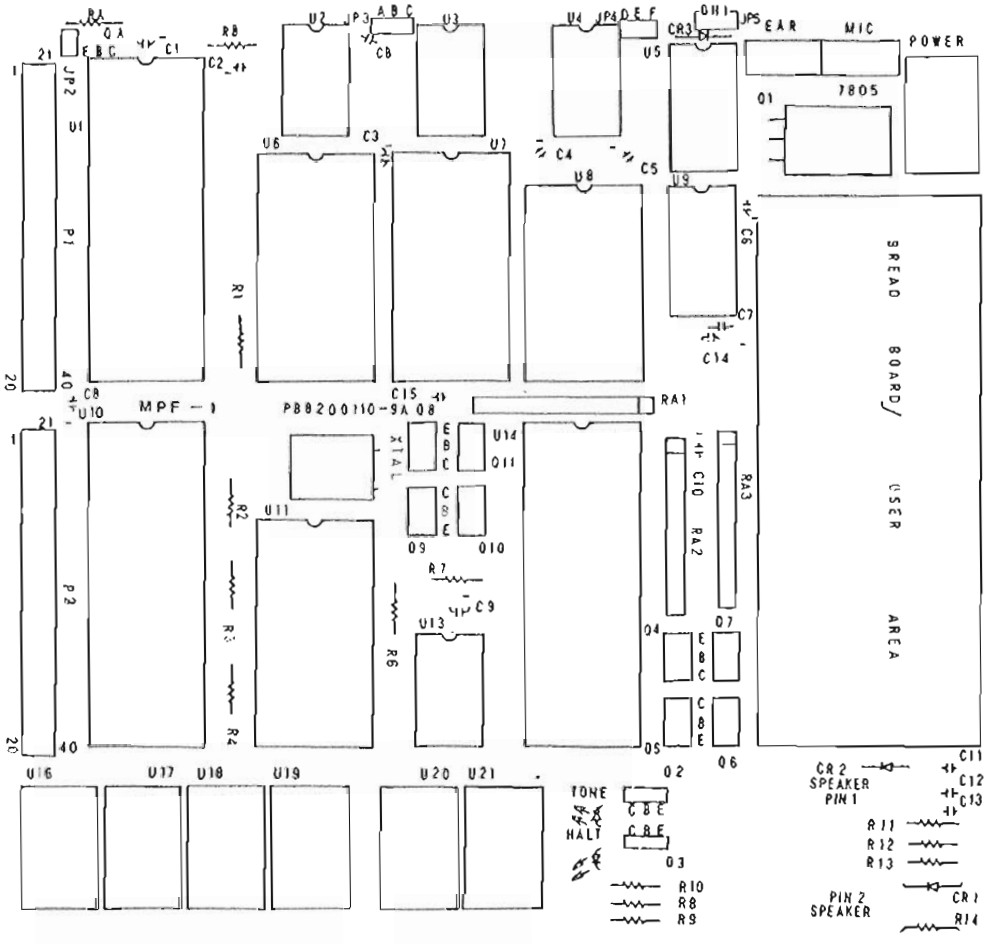
(17) [TAPE WR] : store data from memory onto audio tape.

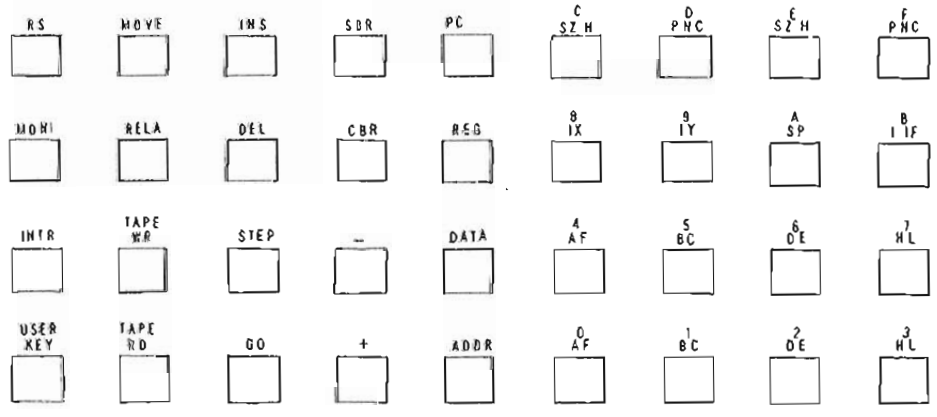
(18) [TAPE RD] : retrieve data from audio tape.

(19) [INTR] : maskable interrupt, connected to CPU's INT pin.

(20) [USER KEY] : user defined key, connected to input port 00, bit 6.

(21) * : [AF 0] [BC 1] [DE 2] [HL 3] [AF' 4] [BC' 5] [DE' 6] [HL' 7] [IX 8] [IY 9] [SP A] [I-IF B] [SZ-H C] [-PNC D] [SZ-H E] [-PNC F]
: hexa-digit or register name.


MPF-I
PB8200110-9A
SPEAKER
PIN 1
PIN 2
SPEAKER
TONE
HALT
EAR
MIC
POWER
7805
BREAD BOARD/ USER AREA


RS
MOVE
INS
SBR
PC
C SZ H
D PNC
E SZ H
F PNC
MONI
RELA
DEL
CBR
REG
8 IX
9 IY
A SP
B IF
INTR
TAPE WR
STEP
DATA
4 AF
5 BC
6 DE
7 HL
USER KEY
TAPE RD
GO
+
ADDR
0 AF
1 BC
2 DE
3 HL

2. General Description

2.1. Functions of Monitor Program

The MPF-I monitor provides the necessary functions for the user to develop his program. These functions include:

(1) The ability to enter the user's program into RAM and to check and modify the program.

(2) Execute the user's program which is stored beginning from the address on the displays.

(3) Using 'Single Step' or 'Set Break Point' function, the user can execute programs step by step or modularly. After each step, control is transferred to the monitor and the current status of the CPU is saved. The user can check or modify registers and memory before executing the next step of the program. This function is very useful in debugging a program.

(4) Other support functions, include audio tape control, and relative address calculation. Using the functions provided by the MPF-I. The user can develop his own special purpose microcomputer system based on MPF-I.

2.2. Notations Used in This Manual

(1) Hexadecimal number system and seven-segment LED display format:

hexadecimal number	decimal number	binary number	seven-segment display
0	0	0000	
1	1	0001	
2	2	0010	
3	3	0011	
4	4	0100	
5	5	0101	
6	6	0110	
7	7	0111	
8	8	1000	
9	9	1001	
A	10	1010	
B	11	1011	
C	12	1100	
D	13	1101	
E	14	1110	
F	15	1111	

Fig. 2-1 Number system

(2) Each display is assigned a number for reference purposes as shown in Fig. 2-2.

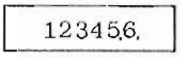


Fig. 2-2 the display number

(3) When the contents of the display are unknown or do not matter an 'X' will be indicated.

(4) A square stands for a key button, as shown! in figure 2-3.

Fig. 2-3 Symbols for buttons

(5) <address> stands for a memory address which is 4 hexadecimal digits entered by the user. If more than four digits are entered, the last four digits are accepted by MPF-I. If less than 4 digits are entered, leading digits are assumed to be 0.

(6) <data> stands for 1 byte of data which is 2 digits entered by the user. The rules are the same as for <address>.

(7) If some key <address>, or<data> is enclosed by '[]',e.g. [<address>], it may be omitted.

2.3. Error Messages:

When an illegal key is entered, the monitor program will blank out the Display to indicate an error condition has occurred. Some program errors will cause an error pattern such as | -Err | to be displayed. When this occurs, locate and enter the appropriate key.

2.4. RAM Addressing

The addresses of the basic RAM are from 1800 to 1FFF. The addresses reserved for expansion RAM are from 2000 to 2FFF. 1F9F - 1FF3 of the basic RAM are allocated to the monitor. The user should read chapter 4 before using this area of memory.

3. Introduction to Operation

This chapter is divided into three parts: basic operations, program debugging, and support functions. Notations are described in 2.2.

3.1. Basic Operations

3.1.1 System Reset Key - [RS]

Pressing the Reset Button will display [UPF--1]

On a power-up, the six digit (UPF--1) are shifted out one-by-one from right to left. The monitor program is initialized either the reset button is pressed or on a power-up.

3.1.2 Examine or modify the contents of memory: [ADDR] and [DATA] Keys

[ADDR] <address>[[DATA] [<data>]] [+] [<data>] [+]

EXAMPLE: Check the contents of memory locations 0000-0003

KEY	DISPLAY	COMMENTS
[ADDR]	X.X.X.X.X X	The four index points notify user to input ADDress.
[AF 0]	0.0.0.0.0 6	The contents of ADDR 0 are 6 Note: Display format for ADDR ADDR = 0. 0. 0. 0. – 0 0 [Address Bytes] [Data Bytes] refer to Description for addition information I-12)
[+]	0 0 0 1 0.0.	Note that the [+] KEY increments the ADDR counter by ONE.
[+]	0 0 0 2 1.0.	Contents of ADDR 0002 is 10.
[+]	0 0 0 3 F. E.	Contents of ADDR 0003 is FE.

EXAMPLE: Change the contents of 1800 into AB, 1801 into CD.

KEY	DISPLAY	COMMENTS
[ADDR]	X.X.X.X.X X [ADDR field]	4 index points in ADDRESS field notify USER to input ADDRESS.
[BC 1] [IX 0] [AF 0] [AF 0]	1.8.0.0.X X	enter 1800 by pressing the appropriate KEYS.
[DATA]	1 8 0 0 X.X. [DATA field]	Pressing the data function KEY allows the user to input data into DATA Field.

[3] [A]	18003A.	Enter 3A into the DATA FIELD.
[B]	1800A.B.	enter B into the Data field. If DATA is more than two digits the last two will be used.
[+]	1801X.X.	ADDR Feild increases by one. The 2 points in the DATA Field notify user to input Data.
[C] [D]	1801C.D.	Enter Data by pressing the C and D Keys.

EXAMPLE: Update the contents of 0000

KEY	DISPLAY	COMMENTS
[ADDR] [0]	0.0.0.0.0 6	ADDRESS is 0000
[DATA]	0 0 0 0 0.6.	
[5]		The contents of ROM cannot be changed so the display is blanked. After releasing the Key the Display will return as before.
	0 0 0 0 0.6.	

[description]:

Addr means address. After pressing this key the display is in the standard format, i.e., the left four digits stand for the address and the right two digits represent the data. The address field is indicated by four points and requires 4 digits. If more than 4 digits are keyed in, only the last 4 are accepted. If less than 4 digits are entered, the 4 hex digits on the display are assumed to be the address.

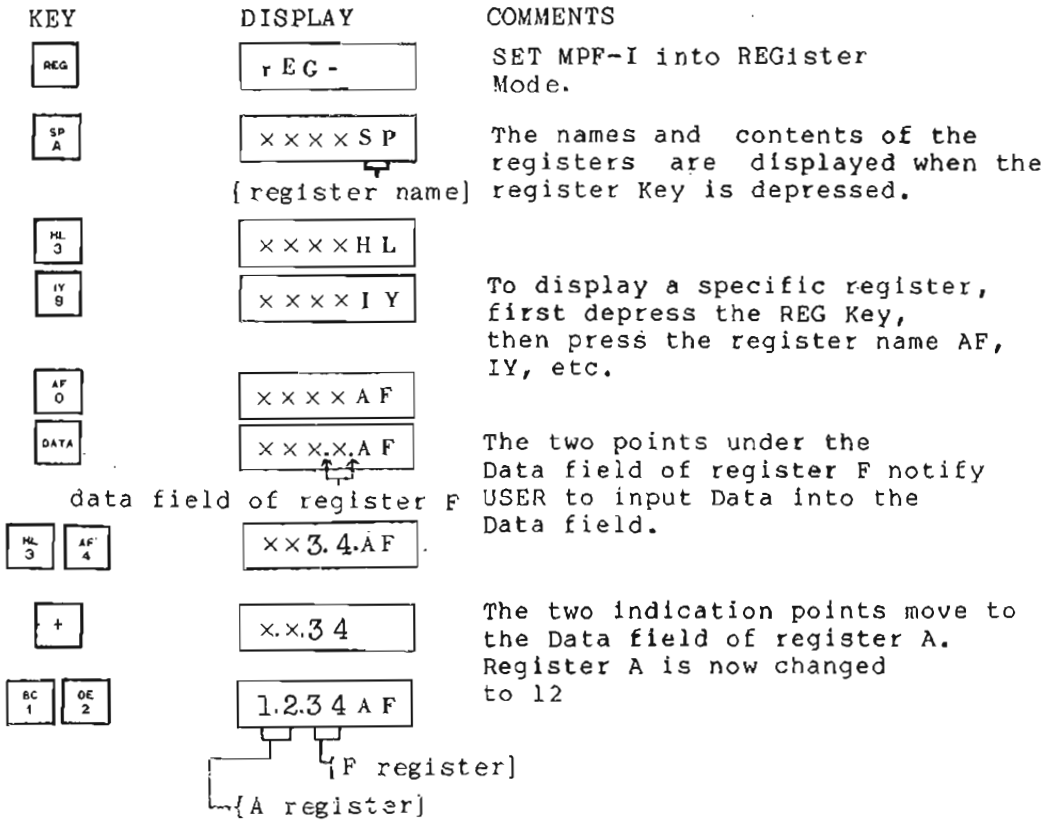
When [DATA] is pressed, the indication points will be shifted to the rightmost two digits prompting the user to enter data. The content of the addressed RAM location will be replaced by the entered data. Pressing [+] or [-] will increase or decrease the address field. If the indication points are already in the data field, then it is unnecessary to press [DATA] . After pressing [ADDR] , the user may press [+] or [-] directly.

If the user attempts to change the contents of ROM, the display will blank out. After releasing the key, the display will be restored.

3.1.3 Examine & Update Registers [REG] and [DATA] Keys

[REG] <register name>[[DATA] [<data>] [+] [<data>] [+] .....

EXAMPLE: Check the contents of SP, HL, IY registers. Change the contents of register A to 12, and register F to 34.



[description]:

<register name> is the name of the register. Each register is addressed by one key. When [REG] is pressed, the display becomes [rEG-] , prompting the user to key in the name. After pressing register name, the right field of the display is the register mnemonic, the left field is the register content. For example, [1234SP] means the contents of the stack pointer is 1234. <register name> is a one key command. If the user wants to check several registers, he just presses <register name> once for each register.

It is necessary to press [DATA] , [+] or [-] if the user wants to update the contents. The change is done on a byte basis. When [DATA] , [+] or [-] is pressed, the display will show two points, notifying the user to input data and indicating that the register is being changed. Pressing [+] or [-] will move the indication points in the direction shown in figure 3-1.

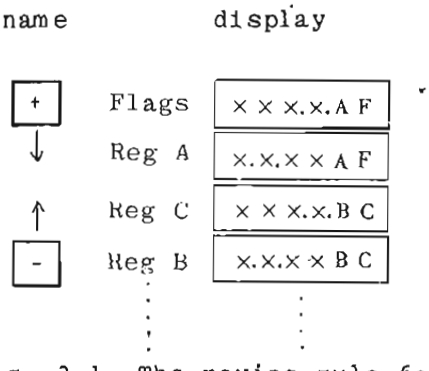
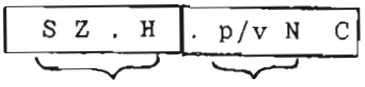


Fig. 3-1 The moving rule for REG indication points

The followings are some special register mnemonics:

1. The alternate registers AF', BC', DE', HL' are indicated by the decimal point at the right.
2. IX, IY is indicated by ㅏ , ㅑ .
3. Register I and interrupt IFF 2 is indicated by IF.

The meaning of each bit in F is shown in figure 3-2.



FH (Flag High) FL (Flag Low)

'.' = does not matter

Fig. 3-2 Flag register

MPF-I decodes the flag register and displays it in 4 bit groups. To display one of four bit groups, refer to the table below

Selection	Key
FH	SZ.H
FL	PNC
FH.	SZ.H'
FL.	PNC'

When decoded flags are modified, only the least significant bit (LSB) of the input key is used. The next time you check AF register, the contents will be updated automatically.

EXAMPLE: Check the carry flag and update it

KEY	DISPLAY	COMMENTS
REG AF 0	××29AF	Contents of F is 29.
PNC D	1001FL	Check the carry bit.
DATA	100.1.FL	
AF 0	100.0.FL	Reset carry flag.
REG AF 0	××28AF	F is updated automatically.

3.1.4 Program Counter - [PC] Key

Reset user's program counter. The basic RAM of MPF-I is 2K bytes. It can be expanded to 4K bytes. When the monitor is reset, it finds the lowest RAM address(1800) and sets the user's program counter to this address. If PC is pressed after RS , the left of the display is the lowest RAM address. See Example A

Example-A

KEY	DISPLAY	COMMENTS
[RS]	UPF--1	
[PC]	1800××	Lowest RAM ADDress.

3.2. Program Debugging

3.2.1 Program Execution — [GO] Key

This key is valid only when the display is in the standard Addr-Data format. After pressing this key, the CPU jumps to the address on the display. Before transferring control to the user's program, it restores all the user's registers. User's registers can be preset by pressing [RS].

EXAMPLE: Executing program

KEY	DISPLAY	COMMENTS
	1FAFSP	The left field of the display is not an address.
[GO]		Display is blank, indicating an error.
	1FAFSP	Display returns to normal after releasing the key.
[PC]	1800××	The left field is an address.
[GO]	××××××	CPU starts execution from 1800.

3.2.2 Single Step - [STEP] Key

[STEP] is similar to [GO]. It is valid only when the display is in Addr-Data form. Pressing this key causes the CPU to execute the instruction pointed to by the current setting of the PC register. After execution, the monitor regains control and displays the new PC and its contents. The user may examine and modify registers and memory contents after each step.

EXAMPLE: Store a program in RAM and execute it by single steps

KEY	DISPLAY	COMMENT
[RS]	U P F - - 1	Reset system.
[PC]	1 8 0 0 ×.×.	Program from 1800. (Z80 instructions)
[HL 3] [SZ·H E]	1 8 0 0 3.E.	
[+] [AF 0]	1 8 0 1 0.0.	LD A,0
[+] [HL 3] [SZ·H C]	1 8 0 2 3.C.	INC A
[+] [AF 4] [HL 7]	1 8 0 3 4.7.	LD B,A
[PC]	1 8 0 0 3.E.	Display is in the Addr-Data form, address is 1800.
[STEP]	1 8 0 2 3.C.	First step, PC becomes 1802.
[REG] [AF 0]	0 0 × × A F	Register A is 0.
[PC] [STEP]	1 8 0 3 4.7.	Second step, PC becomes 1803.
[REG] [AF 0]	0 1 × × A F	Register A incremented.
[PC] [STEP]	1 8 0 4 ×.×.	Third step, PC becomes 1804.
[REG] [BC 1]	0 1 × × B C	Register B becomes 01.

When executing using single step, the monitor uses user's stack to store interrupt return address. The user's stack pointer must point to RAM. If not, [Err-SP] will be displayed. If user's stack pointer points to system stack area, [SYS-SP] will be displayed. Stack overlap will cause an error when 'RET' instruction is executed. In these two cases, you must change the stack pointer or press the reset key. After reset, the system will set user's SP to its default value, the user then need not worry about his stack pointer. (See section 4.5)

3.2.3 Set Break Point - [SBR] Key

When a program is long, single step execution can be very time consuming. Setting break points allows the program to execute more than one instruction and then halt. Pressing step many times has almost the same effect but takes longer. The monitor regains control whenever user's PC passes a specified break point address. The user may examine or modify memory and registers when his program has reached a break point.

SBR means set break point. When the display is in Addr-Data form with address pointing to RAM area, pressing this key causes the displayed address to be set as a break point.

EXAMPLE: Store the following program in RAM. Use SBR to see the results of the execution

address	machine code	instruction
1800	3E00	LD A,0
1802	3C	INC A
1803	47	LD B,A
1804	04	INC B
1805	48	LD C,B
1806	FB	EI

KEY	DISPLAY	COMMENTS
RS	UPF--1	RESET
ADDR BC 1 IX B AF 0 AF 0	1.8.0.0.××	Set Starting ADDR
DATA HL 3 SZ·H E	1800 3.E.	Initialize Data Field
+ AF 0 AF 0	18010.0.	Increment Program Counter
+ HL 3 SZ·H C	18023.C.	Increment Program Counter
+ AF 4 HL 7	18034.7.	Increment Program Counter
+ AF 0 AF 4	18040.4.	Increment Program Counter
+ AF 4 IX 8	180548.	Increment Program Counter
+ ·PNC F I·IF B	1806F.B.	Increment Program Counter
SBR	1.8.0.6.F.B.	Set Breakpoint at 1806
ADDR BC 1 IX B AF 0 AF 0	1.8.0.0.3E	Program starts at 1800
GO	1807×.×.	The program is executed from 1800-1806. The program halts at 1807. Note this is the ADDR of next instruction.
REG	rEG-	To verify results use REG Key
AF 0	01××AF A reg F reg	The contents of the A reg is correct.
BC 1	0202BC B reg C reg	The value in the B register are correct.
I·IF B	0001IF	The interrupt Flip Flop is set. This is the result of EI (Enable Interrupt).
PC -	1.8.0.6.F.B.	Change instruction from EI to DI (Disable interrupt)
·PNC F HL 3	1.8.0.6.F.3.	Enter F3 into Data field.
ADDR BC 1 IX B AF 0 AF 0	1.8.0.0.3E	Set starting ADDR of program.
GO	1807×.×.	Execute
REG I·IF B	0000IF	Check 1FF

Note: Bit has been reset.
(result of DI
instruction)

[Description]:

(1) It is illegal to set break points in ROM area. If you do so, the monitor will blank out the display.

(2) If one instruction has more than one byte, a break point must be set at the first byte. Otherwise, errors will occur.

(3) When the display is in the Addr-Data form and the address field is the break address, six index points are set to indicate that the address is a break point.

(4) The contents of a break address can still be modified by [ADDR] key.

(5) When the user's program executes to the break point, the display is in the Addr-Data form. The address field is the user's PC.

(6) After program executed the break point, all the status and registers are saved.

(7) See 3.2.2 for stack rules.

(8) Only one break point may be set.

3.2.4 Clear Break Point [CBR] Key

If the user want to eliminate the break point in his program, he can press [CBR] to clear the break point. This key is accepted at any time. After pressing it, the display will become [F.F.F.F.F.F.]. (Break point is set to FFFF.)

3.2.5 Immediately Break - [MONI] Key

When executing a program, many errors may occur. For example, a program will lose control when the CPU executes a nonexistant operation code (opcode), or when a program has an infinite loop.

Moni means monitor. Any time you press this key, the same mechanism as used by single step will transfer control to the monitor. Then User's PC and its contents are displayed. When the HALT instruction is executed, pressing [MONI] will return control to monitor and retrieve the contents of the new PC. After [MONI] is pressed, the monitor will check the user's SP. The rules are the same as for single step and break point.

EXAMPLE: HALT and return to MPF-I monitor

KEY	DISPLAY	COMMENTS
[RS] [PC]	1800 x.x.	
[ML 7] [OE 6]	1800 7.6.	Store HALT in 1800.
[GO]		CPU halts, the display is blanked, the LED HALT is turned on.
[MONI]	1801 x.x.	The display is of the Addr-Data form. The address field is the user's PC. All registers are reserved.

EXAMPLE: Pressing this key when monitor is being executed

KEY	DISPLAY	COMMENTS
[RS]	UPF--1	The monitor is scanning the keyboard. The system treats the monitor as the user's program. The user's SP is in the system stack.
[MONI]	SYS-SP	

3.3. Support Functions

3.3.1 Block Transfer - [MOVE] Key

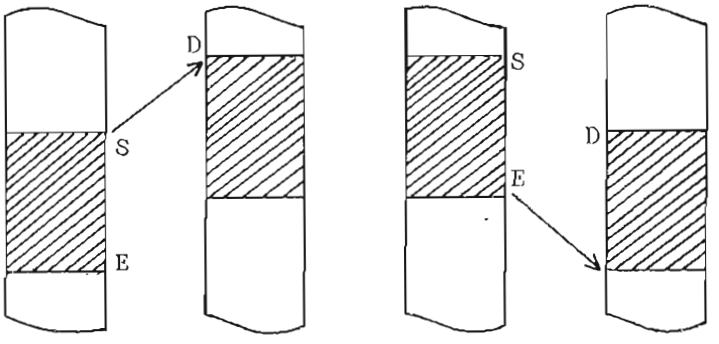
[MOVE] <address> [+] <address> [+] <address> [GO]

EXAMPLE: Move the data in 1800 - 18FF to 1810 - 190F

KEY	DISPLAY	COMMENTS
[MOVE]	x.x.x.x.- S	S is the mnemonic of starting address.
[BC 1] [IX 8] [AF 0] [AF 0]	1.8.0.0.- S	Starting address = 1800
[+]	x.x.x.x.- E	E is the mnemonic of ending address.
[BC 1] [IX 8] [PNC F] [PNC F]	1.8.F.F.- E	Ending address = 18FF
[+]	x.x.x.x.- d	D is the mnemonic of destination address.
[BC 1] [IX 8] [BC 1] [AF 0]	1.8.1.0.- d	Destination address = 1810
[GO]	1 8 1 0 x.x.	Transfer completed, the last byte moved is 1810

[Description]:

After pressing the [MOVE] key, the display becomes [××××-S]. S means the starting address of the data to be transfered.. After pressing [.], the display becomes [××××-E], E means the ending address of the data to be transfered. Press [.] again and the display becomes [××××-D]. D means the destination address of the data to be moved. When finished, the display is of the Addr-Data form. The address field is the last byte moved. Movement can be upward or downward. When moving upward, the last address is the lower limit of the destination area. When moving downward, the last address is the upper limit of the destination area, as shown in figure 3-3. Because of the fast speed of the microcomputer, the transfer can be finished instantaneously. After pressing [GO] the result will be displayed at once.



moving upward moving downward

Fig. 3-3 Function of [MOVE]
(arrow indicates the first byte moved)

If the destination area overlaps the system stack, the system stack will be destroyed. The user should pressed [RST] to reset the system.

3.3.2 Data Deletion - [DEL] Key

This key is valid when the display is of the Addr-Data form. Pressing this key causes the data of the displayed address to be deleted. All the data above this address is shifted down one position.

EXAMPLE: Assume the present contents of RAM and the desired contents are as follows:

	ADDRESS	OLD DATA	DATA AFTER DELETING
	1800	00	00
	1801	11	11
delete address->	1802	11	22
	1803	22	33
	1804	33	44
	1805	44	XX

KEY	DISPLAY	COMMENTS
[ADDR -] [BC 1] [IX 8] [AF 0] [DE 2]	1.8.0.2.1 1	To change the display to the Addr-Data form and enter the address to be deleted.
[DEL]	1 8 0 2 2.2.	The old contents of 1802 have been deleted and data above it have been shifted down. The new contents of 1802 are 22, which was the original contents of 1803.
[ADDR] [BC 1] [IX 8] [AF 0] [AF 0]	1.8.0.0.0 0	Check.
[+]	1 8 0 1 1.1.	
[+]	1 8 0 2 2.2.	
[+]	1 8 0 3 3.3.	
[+]	1 8 0 4 4.4.	

[Description]:

Data in ROM can not be deleted. The valid regions for this key are 1800 - 1DFF. When the deleted address is between 1800 - 1DFF, all the data after this address shift down position. The last one (1DFF) is filled with 0.

3.3.3 Data insertion - [INS] Key

[INS] <data>

When the display is of the Addr-Data form, the input data will be inserted after the displayed address.

EXAMPLE: Assume the contents of RAM are as follows:

	ADDRESS	OLD DATA	DATA AFTER INSERTION
	1800	00	00
	1801	11	11
	1802	22	22
insert 33 here ->	1803	44	33
	1804	55	44
	1805	66	55

KEY	DISPLAY	COMMENTS
[ADDR] [BC 1] [IX 0] [AF 0] [DE 2]	1.8.0.2.2 2	To change the display to the Addr-Data form and enter the address of the insertion. Insert one byte after 1802, address field becomes 1803. Key in data 33.
[INS]	1 8 0 3 0.0.	
[HL 3] [HL 3]	1 8 0 3 3.3.	
[ADDR] [BC 1] [IX 8] [AF 0] [AF 0]	1.8.0.0.0 0	Check
[+]	1 8 0 1 1.1.	
[+]	1 8 0 2 2.2.	
[+]	1 8 0 3 3.3.	
[+]	1 8 0 4 4.4.	
[+]	1 8 0 5 5.5.	

[Description]:

The valid region for this key is the same as [INS]. After insertion, the last byte of the inserted block is lost.

The inserted address is one byte after the displayed address. Pressing this key causes all the data after the displayed address to be shifted up one position. Then the address field is incremented by one and the user may enter the data he wants to insert.

3.3.4 Relative Address Calculation - [RELA] Key

Instructions JR and DJNZ require relative addresses. MPF-I supports the calculation of relative addresses throught the [RELA] key.

[RELA] <address> [+] <address> [GO]

EXAMPLE: Assume there is a JR instruction in your program. The address of opcode is 1800, the address to jump to is 1804.

KEY	DISPLAY	COMMENTS
[RELA]	X.X.X.X.- S	S is the mnemonic of starting address.
[BC 1] [IX B] [AF 0] [AF 0]	1.8.0.0.- S	Starting address = 1800
[+]	X.X.X.X.- d	D is the mnemonic of destination address.
[BC 1] [IX B] [AF 0] [AF 4]	1.8.0.4.- d	Destination = 1804
[GO]	1 8 0 1 0 2.	MPF-I computes the relative address and stores the result in the next byte of the JR opcode. The result is also displayed.

[Description]:

After pressing [RELA], the display becomes [X.X.X.X.- S]. S represents the starting point of JR or DJNZ. Pressing [.], the display becomes [X.X.X.X.- D]. d represents the destination address of JR or DJNZ. Pressing [GO], MPF-I computes the relative address then stores it in the 2nd byte of opcode. The display becomes of the Addr-Data form. The address containing the relative address is displayed. If the result exceeds decimal +127 or -128, the display becomes [- E r r].

3.3.5 Storing Data onto Tape - [TAPE WR] Key

Cassette tape is a large capacity non-volatile storage medium. MPF-I contains hardware and software drivers.

[TAPE WR] <file name> [+] <address> [+] <address> [GO]

EXAMPLE: store the data of 1800 -18FF on tape, use 1234 as file name.

KEY	DISPLAY	COMMENTS
[TAPE WR]	x.x.x.x.- F	F is the mnemonic of filename.
[BC 1] [DE 2] [HL 3] [AF 4]	1.2.3.4 - F	filename = 1234
[+]	x.x.x.x.- S	S is the mnemonic of starting address.
[BC 1] [IX 8] [AF 0] [AF 0]	1.8.0.0.- S	Starting address = 1800
[+]	x.x.x.x.- E	E is the mnemonic of ending address.
[BC 1] [IX 8] [PNC F] [PNC F]	1.8.F.F.- E	Ending address = 18FF
(PLAY & REC)		Connect the microphone of the tape recorder to MPF-I MIC. Start recording by pressing PLAY and REC key of recorder.
[GO]		Begin to output data. During transfer, the display is dark, but the TONE-OUT LED is on.
	1 8 F F x.x.	When transfer is completed, the ending address is displayed.

[Description]:

Pressing [TAPE WR], the display becomes [X.X.X.X.- F]. F means file name. It is used to distinguish different data sets stored on a single cassette. It is also used to read back data. Press [·] and the display becomes [X.X.X.X.- S]. S represents the starting address of the data to be written. Press [·] again and the display becomes [X.X.X.X.- E]. E represents the ending address of the data to be written. Before pressing [∞], you must connect the microphone of the recorder to MIC jack of MPF-I and press PLAY and REC to start recording. If the recorder is not ready and you press [∞], data is still sent out. This data will not be recorded on tape. During transfer the display is blank, the TONE-OUT LED is on and a tone sounds.

3.3.6 Reading Data from Tape - [TAPE RD] Key

[TAPE RD] <filename> [GO]

EXAMPLE: Read the data from a tape recorder, filename is 1234, the data on the tape was written by using the [TAPE WR] key, see storing data onto tape.

KEY	DISPLAY	COMMENTS
[TAPE RD]	X.X.X.X.- F	F is the mnemonic of filename.
[BC 1] [DE 2] [HL 3] [AF 4]	1.2.3.4.- F	Filename = 1234
		Connect the recorder (using earphone jack) to the EAR jack in MPF-I.
[GO]		Start execution. The display is blank while MPF-I is searching for the filename.
(PLAY)		Press PLAY on the recorder. The recorder output volume should be turned to maximum. MPF-I echoes the signal read from tape on its own speaker (if the volume is too low, then there will be no sound). Every file name read by the monitor will be displayed for 1.5 seconds. When the desired file is found, '.' is changed into '-'. When finished, the last address read in is displayed.
	. . . . .	
	1234-F	

	18FFX.X.	

[Description]:

Before execution, the user must connect the recorder (using earphone jack) to the EAR jack in MPF-I. Turn the volume of the recorder to maximum. Then press [↵], and finally, start the recorder (PLAY). Initially, the display is [• • • • • •].When the desired file is found, the display becomes [- - - - - -].

Starting and ending addresses are already stored on the tape so there is no need to input them. The user just needs to input the file name. A check is also recorded on the tape which MPF-I will check when reading back. If not matched, the display will be [- E r r].If matched, the last input byte will be displayed.

If the data read from the tape is stored in a system stack, errors will occur. Care must be taken when you prepare tape data by ****. The tape data is echoed on the MPF-I speaker, so it is very easy to determine whether the tape is empty or not. This allows you to check a tape before recording data on it, so you do not destroy data that has been previously recorded.

4. Software and Hardware Description

4.1. Memory Address

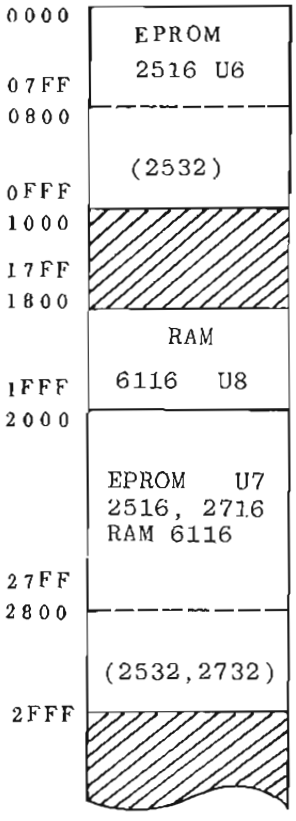


Fig 4-1 Memory map

[Description]:

(1) U6 EPROM: monitor

U7 RAM or EPROM: reserved for expansion

U8 RAM: basic RAM of which 1FAF-1FFF are used by monitor

(2) Address lines are fully decoded in MPF-I. Traces don't need to be cut or jumpers added on the PC board if 2516, 2716, or 2532 are inserted in U7.

a. The following lines need to cut and jumpered if a 2732 is inserted in U7.

Cut lines	Jumped lines
PIN 1,2 of jumper	PIN 2,3 of jumper
PIN 3,4 of jumper	PIN 4,5 of jumper
PIN 5,6 of jumper	PIN 6,7 of jumper

b. The following lines need to cut and jumpered if a 6116 is inserted in U7.

Cut lines	Jumped lines
PIN 3,4 of jumper	PIN 4,5 of jumper

4.2. Input/Output (I/O)Address)

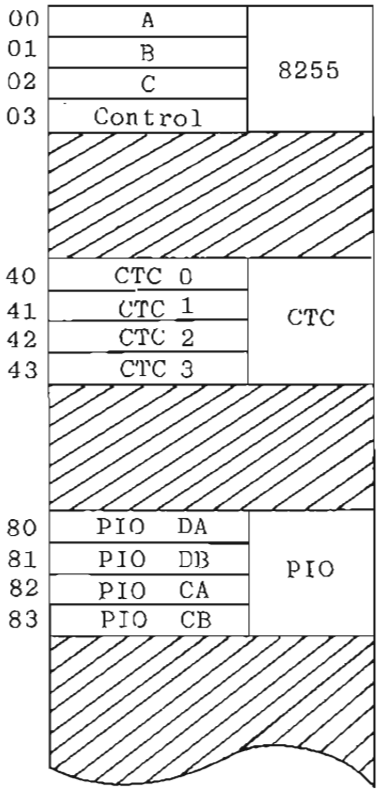


Fig. 4-2 I/O address map

[Description]:

(1) The 8255 is a programmable peripheral interface with 24 parallel I/O lines. These 24 I/O lines are divided into three 8-bit ports. (See 8255 data sheet for details).

(2) The control word of 8255 is 03. Port A is an input port, ports B and C are output ports.

(a) Port A (address 00):
bit 7: tape input,
bit 6: connected to [USER KEY] key, active low,
bit 5 - 0: connected to 6 rows of the keyboard matrix The input signal becomes low only when keys in the active column are pressed.

(b) Port B (address 01) controls the seven segments and decimal point of the display. Figure 4-3 shows the name of each segment and the corresponding bit in port B. All output bits are active high.

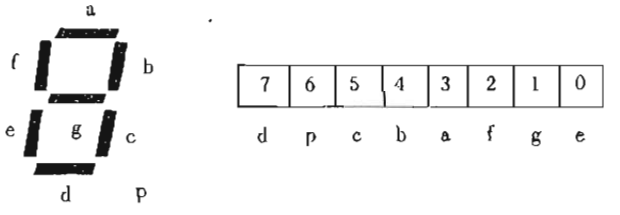


Fig. 4-3 The corresponding bits of the 7-segment display

(c) Port C (address 02):
bit 7: tape output; also connected to the speaker and the LED TONE-OUT. LED is turned on when output is 0.
bit 6: monitor break control. Any attempt to change this bit is forbidden.
bit 5 - 0: is connected to 6 columns of display & keyboard matrix. Bit 0 is the rightmost display, bit 5 is the leftmost display. All these bits are active high.

(3) The Z80 Counter Timer Controller (CTC) is a programmable component with four independent channels that provide counting and timing function for microcomputer systems based on the Z80-CPU.
The I/O addresses of CTC are from 40H to 43H.

(4) The Z80 parallel I/O (PIO) is a programmable, two port device which provides a TTL compatible interface between peripheral devices and the Z80-CPU.
The I/O address of PIO are from 80H to 83H.

(5) Address lines are not fully decoded only A0, A1, A6 and A7 are used. A2 through A5 are undecoded lines.

4.3. Program Interrupt

The nonmaskable interrupt is used by the monitor. The user is not allowed to use it. Pin 16 of the CPU (INT) is connected to jumper I on the left edge of the PC board and to [INTR] . When the monitor code at address 0038 is executed, control will be transfered to the address stored in 1FFE & 1FFF. During this process, all CPU status are an affected. The default contents of 1FFE & 1FFF are 0066. This is the entry point of the service routine. 0038 is executed in the following situations:

(1) Mode 1 interrupt is acknowledged;

(2) Instruction 'RST 38H' (opcode FF) is executed;

(3) The data bus are pulled high. If mode 0 interrupt is acknowledged without the interrupt vector, RST 38H will be executed.

(4) When there is an error in program execution and jumps to a nonexistent memory. The opcode fetched by CPU is FF.

If the contents of 1FEE & 1FEF are not changed after power on, the effect of executing 0038 is the same as for pressing [INT] key or break point. The user may define his own service routine by changing the contents of 1FEE & 1FEF.

4.4. Software Break-Introduction RST 30H(Opcode F7)

RST 30H has the same effect as break. It is called software break because no hardware operation is involved.

It is usually used as the terminator of a user's program. It is also very useful in multi-break-point program debugging.

4.5. Stack

Figure 4-4 shows the stack configuration. The default value of the user's stack pointer is 1F9F. Each time the user's program breaks, the monitor checks his SP. SYS-SP will be displayed if the user's SP points to the system stack. If there is stack related instruction (e.g. RET) in the user's program, an error may occur when user's stack and system stack overlap.

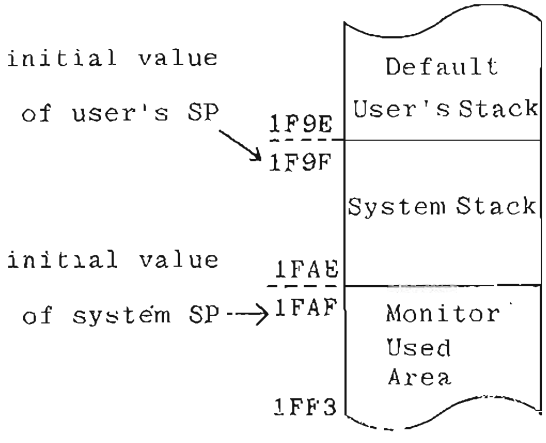


Fig. 4-4 Stack distribution map

SYS-SP can occur in the following situations:

(1) Pressing [NEW] when monitor is controlling the CPU. This operation will destroy all user's registers and should be avoided.

(2) Executing the monitor itself by pressing [SETP]

4.6. Reset

There are two possible results. When the monitor is reset,

(1) Power on

(a) Disable interrupt (IFF set to 0);
(b) I regiser set to 0;
(c) Interrupt mode set to 0;
(d) User's PC is set to 1800.
(e) User's SP is set to 1F9F;
(f) Break point is disabled.
(g) Set the content of 1FFE to 66 and set the content of 1FFF to 00. When the code beginning at 0038 is executed the CPU will jump to 0066. This is equivalent to press [NMI] .
(h) MPF-I is displayed one character at at a time from right to left.

(2) Press [RS] .

(a) - (e) are the same as (1). The contents of 1FFE & 1FFF and break point are unaffected. 'UPF--1' is displayed, (all digits) simultaneously.

4.7. Tape Data Format

(1) Bit format:

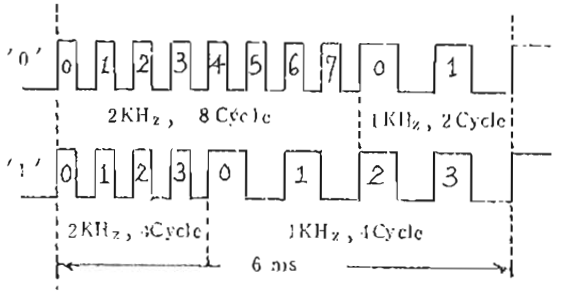


Fig. 4-5 Tape bit format

(2) Byte format:

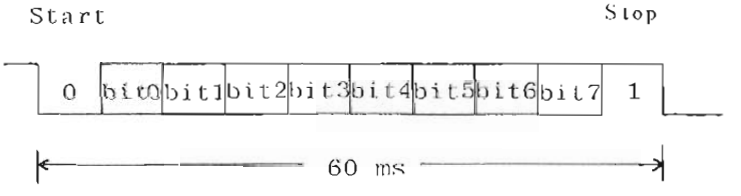


Fig. 4-6 Tape byte format

(3) File format:

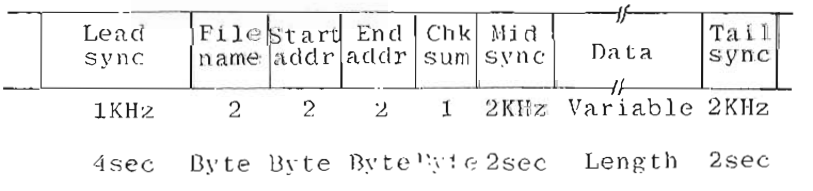


Fig. 4-7 Tape file format

5. Monitor Subroutines

5.1. Summary

ADDRESS	MNEMONIC	FUNCTION
0624	SCAN1	Scan keyboard and display one cycle.
05FE	SCAN	Scan keyboard and display until a new key-in.
0689	HEX7	Convert a hexadecimal digit into the 7-segment display format.
0678	HEX7SG	Convert two hexadecimal digits into 7-segment display format.
05F6	RAMCHK	Check if the given address is in RAM.
05E4	TONE	Generate sound.
05DE	TONE1K	Generate sound at 1K Hz.
05E2	TONE2K	Generate sound at 2K Hz.

5.2. SCAN1

[Address]: 0624

[Function]: Scan keyboard and 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;

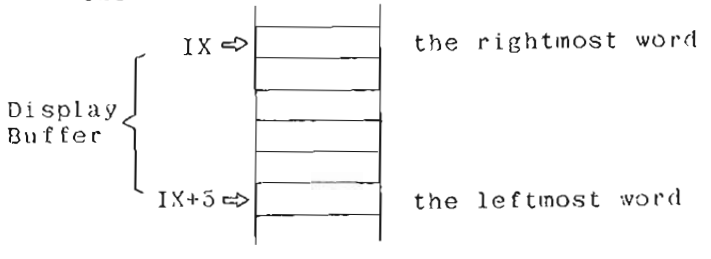
(2) If key-in, carry flag =0 and the position-code of the key is stored in register A. (See appendix A.)

[Register]: Destroy AF, A'F', B'C', D'E'

[Supplement]:

(1) 6 bytes are required for storing 6 word patterns.

(2) IX points to the rightmost word, IX+5 points to the leftmost word.



(3) See appendix A for the relation between each bit and the seven segments.

5.3. 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 (see appendix A).

[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'.

5.4. HEX7

[Address]: 0689

[Function]: Convert a hexadecimal number into its 7-segment display format.

[Input]: The least significant 4 bits of register A contain the hexdecimal number (0-F).

[Output]: The result is also stored in register A.

[Register]: Destroy AF only.

5.5. HEX7SG

[Address]: 0678

[Function]: Convert two hexa-decimal numbers into a 7-segment display format.

[Input]: The first number is stored in the right 4 bits of A. The second number is stored in the left 4 bits of A.

[Output]: The first display pattern is stored in (HL), the second is in (HL+1), HL is increased by 2.

[Register]: Destroy AF, HL.

5.6. RAMCHK

[Address]: 05F6

[Function]: Check if the given address is in RAM.

[Input]: The address to check is in HL.

[Output]: If it is in RAM then Zero flag = 1, otherwise Zero flag = 0.

[Register]: Destroy AF

5.7. TONE

[Address]: 05E4

[Function]: Generate sound.

[Input]: (1) C controls the frequency of the sound.
The period is about (44+Cx13)x2x0.56 micro-sec,
and the frequency is 200/(10+3xC) KHz.
(2) HL contains the number of cycles. (max. value
is 32768).

[Output]: None

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

5.8. TONE1K

[Address]: 05DE

[Function]: Generate a sound of 1KHz.

[Input]: Number of periods is in HL.

[Output]: None.

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

5.9. TONE2K

[Address]: 05E2

[Function]: Generate a sound of 2KHz.

[Input]: The number of periods is in HL.

[Output]: None

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

5.10. Program Examples

EXAMPLE 1: Display 'HELPUS', HALT when [STEP] is pressed.

```
                        1  ;DISPLAY 'HELP US' UNTIL STEP-KEY PUSHED:
1800                    2          ORG     1800H
1800  DD212018        3          LD      IX,HELP
1804  CDFE05          4  DISP    CALL    SCAN
1807  FE13            5          CP      13H         ;KEY-STEP
1809  20F9            6          JR      NZ,DISP
180B  76              7          HALT
                        8  ;
1820                    9          ORG     1820H
1820  AE               10  HELP    DEFB    0AEH        ;'S'
1821  B5               11          DEFB    0B5H        ;'U'
1822  1F               12          DEFB    01FH        ;'P'
1823  85               13          DEFB    085H        ;'L'
1824  8F               14          DEFB    08FH        ;'E'
1825  37               15          DEFB    037H        ;'H'
                       16  ;
                       17  SCAN    EQU     05FEH
                       18          END
```

Details of the display buffer are given below:

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	a,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

Please refor to Appendix A

EXAMPLE 2: Flash 'HELP US'

Use routine SCAN1 to display 'HELPUS' and blank alternately. Display each pattern 500ms by looping SCAN1 50 times.

```
                        1   ;FLASH 'HELP US':
1800                    2           ORG     1800H
1800  212618              3           LD      HL,BLANK
1803  E5                4           PUSH    HL
1804  DD212018          5           LD      IX,HELP
1808  DDE3              6   LOOP    EX      (SP),IX
180A  0632              7           LD      B,50
180C  CD2406            8   HELFSEG CALL    SCAN1
180F  10FB              9           DJNZ    HELFSEG
1811  18F5             10           JR      LOOP
                       11   ;
1820                   12           ORG     1820H
1820  AE               13   HELP    DEFB    0AEH        ;'S'
1821  B5               14           DEFB    0B5H        ;'U'
1822  1F               15           DEFB    01FH        ;'P'
1823  85               16           DEFB    085H        ;'L'
1824  8F               17           DEFB    08FH        ;'E'
1825  37               18           DEFB    037H        ;'H'
1826  00               19   BLANK   DEFB    0
1827  00               20           DEFB    0
1828  00               21           DEFB    0
1829  00               22           DEFB    0
182A  00               23           DEFB    0
182B  00               24           DEFB    0
                       25   ;
                       26   SCAN1   EQU     0624H
                       27           END
```

The content of 180B determines the flash frequency. You may change it to any value.

EXAMPLE 3: 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  EQU     0678H
                       19           END
```

When a key is pressed, the internal code for that command is displayed in the data field. The user may compare it with Appendix A.

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

```
                        1   ;DISPLAY POSITION CODE
1800                    2           ORG     1800H
1800  DD210019        3           LD      IX,OUTBF
1804  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
```

EXAMPLE 4: Convert 3 continuous bytes into 7-segment display format. Store the results in 1903 ~ 1908 then display them.

```
                        1   ;DISPLAY 3 BYTES IN RAM TO 6 HEXA-DIGITS
1800                    2           ORG     1800H
1800  110019          3           LD      DE,BYTE0
1803  210319          4           LD      HL,OUTBF
1806  0603            5           LD      B,3
1808  1A              6   LOOP    LD      A,(DE)
1809  CD7806          7           CALL    HEX7SG
180C  13              8           INC     DE
180D  10F9            9           DJNZ    LOOP
                       10   ;CONVERSION COMPLETE, BREAK FOR CHECK
180F  DD210319       11           LD      IX,OUTBF
1813  CDFE05         12           CALL    SCAN
1816  76             13           HALT
                       14   ;
1900                   15           ORG     1900H
1900  10             16   BYTE0   DEFB    10H
1901  32             17           DEFB    32H
1902  54             18           DEFB    54H
1903                   19   OUTBF   DEFS    6
                       20   ;
                       21   HEX7SG  EQU     0678H
                       22   SCAN    EQU     05FEH
                       23           END
```

The three bytes of binary data are stored in 1900 - 1902. The user can set a break point at 180F to check if the conversion is correct before displaying the result.

EXAMPLE 5: Simulate a police car siren

The sound of a police car siren is simulated by alternating two different frequencies. Register C controls the frequency of the sound and register pair HL controls the length of the sound.

```
                        1   ;POLICE CAR SIREN:
1800                    2           ORG     1800H
1800  0E00               3   LOOP    LD      C,0
1802  21C000            4           LD      HL,0C0H
1805  CDE405            5           CALL    TONE
1808  0EC0              6           LD      C,0C0H
180A  210001            7           LD      HL,100H
180D  CDE405            8           CALL    TONE
1810  18EE              9           JR      LOOP
                       10   ;
                       11   TONE    EQU     05E4
                       12           END
```

(1) Low frequency: C=0 (equivalent to 256), HL=C0H (192), so the period is (44+13x256)x2x0.56= 3777 micro-sec.

frequency: 1/3777= 265Hz

length of the sound: 3777 micro-sec x192= .73sec

(2) High frequency: C=C0H (192), HL=100H (256), so the period is (44+13x192)x2x0.56= 2845 micro-sec.

frequency: 1/2845=352Hz

length of the sound: 2845 micro-sec x256= .73sec

6. Memory Check

6.1. Check EPROM 0000-07FF

The sum of all monitor codes is zero. Routine ROMTEST at 06A6 uses this property to check the monitor EPROM.

```
                ROMTEST:
06A6    210000          LD      HL,0
06A9    010008          LD      BC,800H
06AC    CD3105          CALL    SUM
06AF    2801            JR      Z,SUMOK
06B1    76              HALT                    ;IF ERROR
06B2    C7      SMUOK   RST     0               ;DISPLAY 'UPF--1'
                ;


0531    AF      SUM     XOR     A
0532    86      SUMCAL  ADD     A,(HL)
0533    EDA1            CPI
0535    EA3205          JP      PE,SUMCAL
0538    B7              OR      A
0539    C9              RET
```

This program calculates the sum of all EPROM codes. If the result is 00, 'UPF--1' is displayed.

The key sequence is as follows:

(1) Correct: the display is `UPF--1`

(2) Error: HALT LED will come on.

6.2. Check RAM Region 1800-1FFF

```
            RAMTEST:
0694        210018              LD      HL,1800H
0697        010008              LD      BC,800H
069A        CDF605      RAMT    CALL    RAMCHK
069D        2801                JR      Z,TNEXT
069F        76                  HALT                ;IF ERROR
06A0        EDA1        TNEXT   CPI
06A2        EA1E07              JP      PE,RAMT
06A5        C7                  RST     0           ;DISPLAY 'UPF--1'
                        ;

            RAMCHK:
05F6        7E                  LD      A,(HL)
05F7        2F                  CPL
05F8        77                  LD      (HL),A
05F9        7E                  LD      A,(HL)
05FA        2F                  CPL
05FB        77                  LD      (HL),A
05FC        BE                  CP      (HL)
05FD        C9                  RET
```

This program tests every byte in region 1800 - 1FFF. If the byte is good, it continues testing till all bytes have been tested. If there is any bad byte, the HALT LED will come on. You can press [RS], [ADDR], [5] to get the address that has the error. Then press [ADDR] to get the content of that byte. If you want to continue testing, you may press [PC] [GO]. The key sequence is as follow:

[RS] [ADDR] [DE 6] [IY 9] [AF 4] [GO]

(1) Correct: [U P F - - 1] is displayed.

(2) Error: HALT LED will come on.

APPENDIX

A. Display format, position-code and internal-code

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

DISPLAY FORMAT :

CODE	BD	30	9B	BA	36	AE	AF	38	BF	8E	3F	A7	8D	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	B1	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	87	B5	B7	A9	07	B6	8A	83	A2	32	02	C0	00
DATA	S	T	U	V	W	X	Y	Z	(	)	+	-	,	
DISP	S	T	U	V	W	X	Y	Z	(	)	+	-	,	

1. Position-Code (CALL SCANI) :

1E SBR	18 CBR	12 '0'	0C '1'	06 '2'	00 '3'
1F '-'	19 PC	13 '4'	0D '5'	07 '6'	01 '7'
20 DATA	1A REG	14 '8'	0E '9'	08 'A'	02 'B'
21 '+'	1B ADDR	15 'C'	0F 'D'	09 'E'	03 'F'
22 INS	1C DEL	16 GO	10 STEP	0A	04
23 MOVE	1D RELA	17 TPWR	11 TPRD	0B	05

2. Internal-Code (CALL SCAN) :

15 SBR	1A CBR	00 '0'	01 '1'	02 '2'	03 '3'
11 '-'	18 PC	04 '4'	05 '5'	06 '6'	07 '7'
14 DATA	1B REG	08 '8'	09 '9'	0A 'A'	0B 'B'
10 '+'	19 ADDR	0C 'C'	0D 'D'	0E 'E'	0F 'F'
16 INS	17 DEL	12 GO	13 STEP	22	20
1C MOVE	1D RELA	1E TPWR	1F TPRD	23	21

APPENDIX B

Theory of Hardware Circuit

A. System Clock

U3a, U3b, and 3.58M Hz crystal produce 3.58MHz signal. This signal is sent to U2a pin 3 to produce 3.58MHz÷2 = 1.79MHz system clock.

B. Reset Signal

U2b is used to trim the Reset signal produced by power on or pressing [RST] key. The trimmed $\overline{RST}$ is sent to CPU and CTC. RST is sent to the 8255.

C. Memory Addressing

MREQ	A15	A14	A13	A12	A11	A10 --- A0	Selected Chip	Address
0	0	0	0	0	X	X --- X	U6	0000-0FFF
0	0	0	1	0	X	X --- X	U7	2000-2FFF
0	0	0	0	1	1	X --- X	U8	1800-1FFF

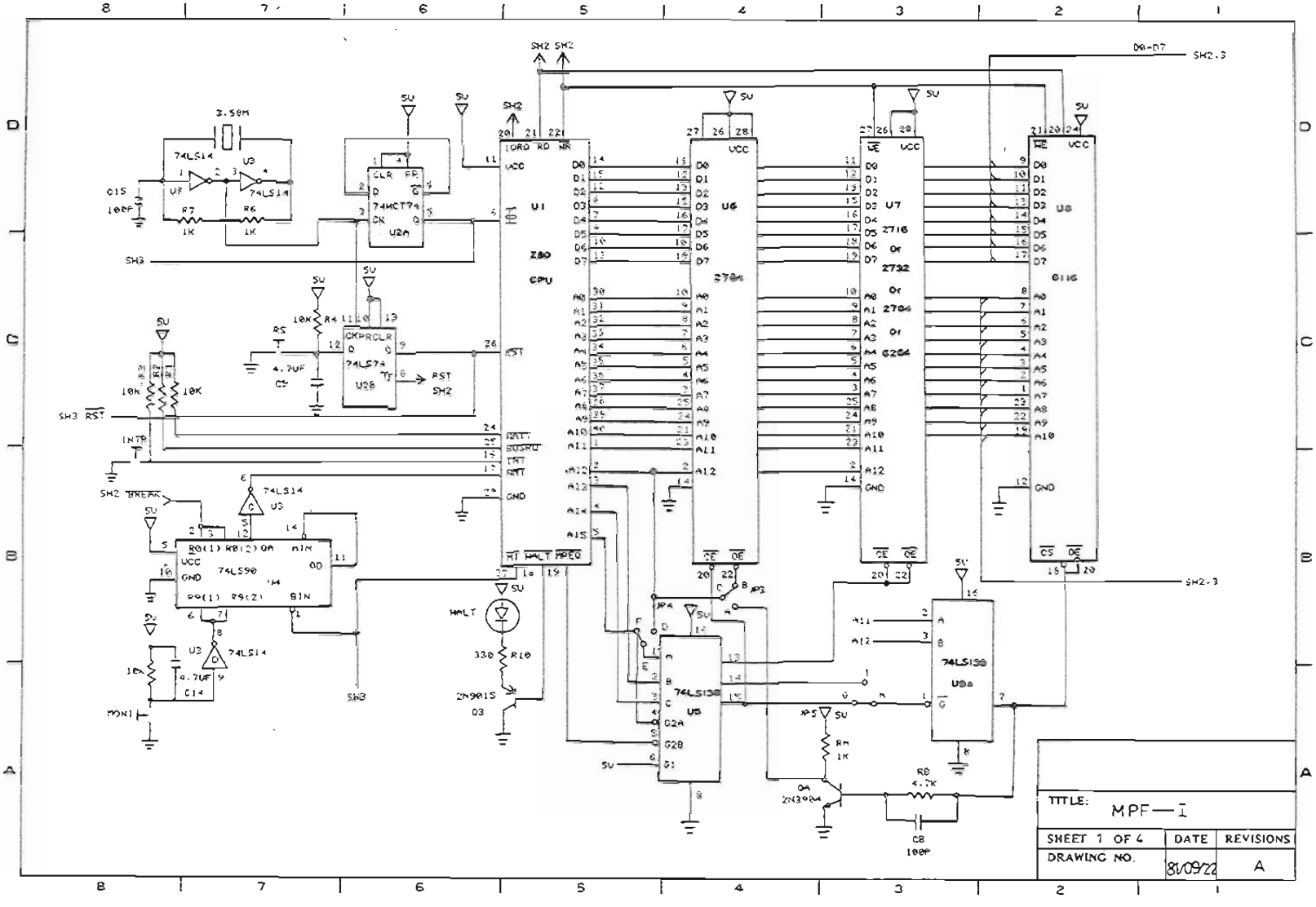
U6 is the monitor for MPF-I, it may be a TMS2516, or an Intel 2716. U7 is a spare socket for future expansion usage, it may be a RAM or a ROM, Circuit design is default for 2716, 2516, 2532(EPROM) when user intends to plug in Intel 2732, or HM 6116(RAM), he should consult the note on Sheet 4 of the schematic. U8 is a system RAM, the memory size is 2K bytes.

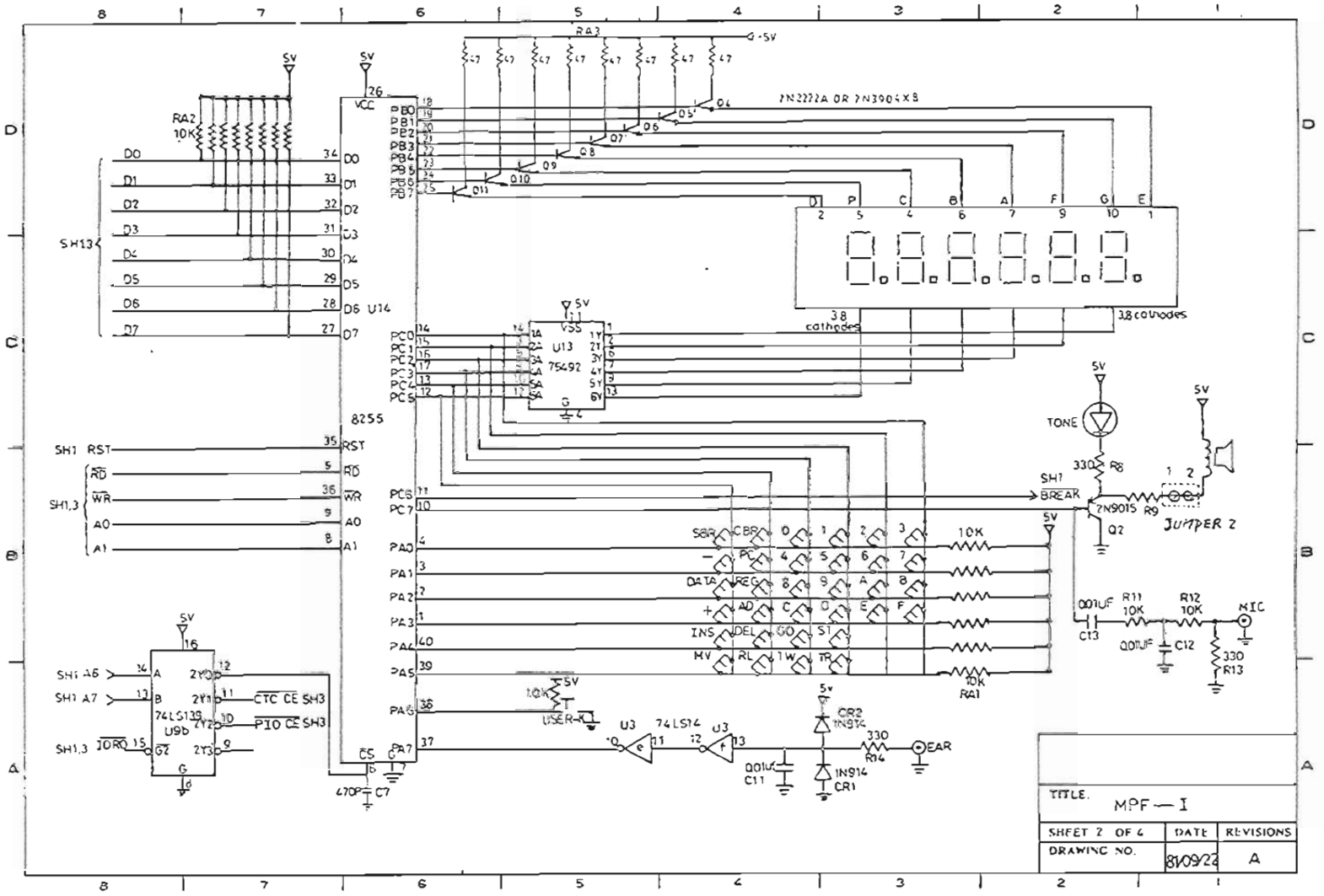
D. Input/Output port addressing

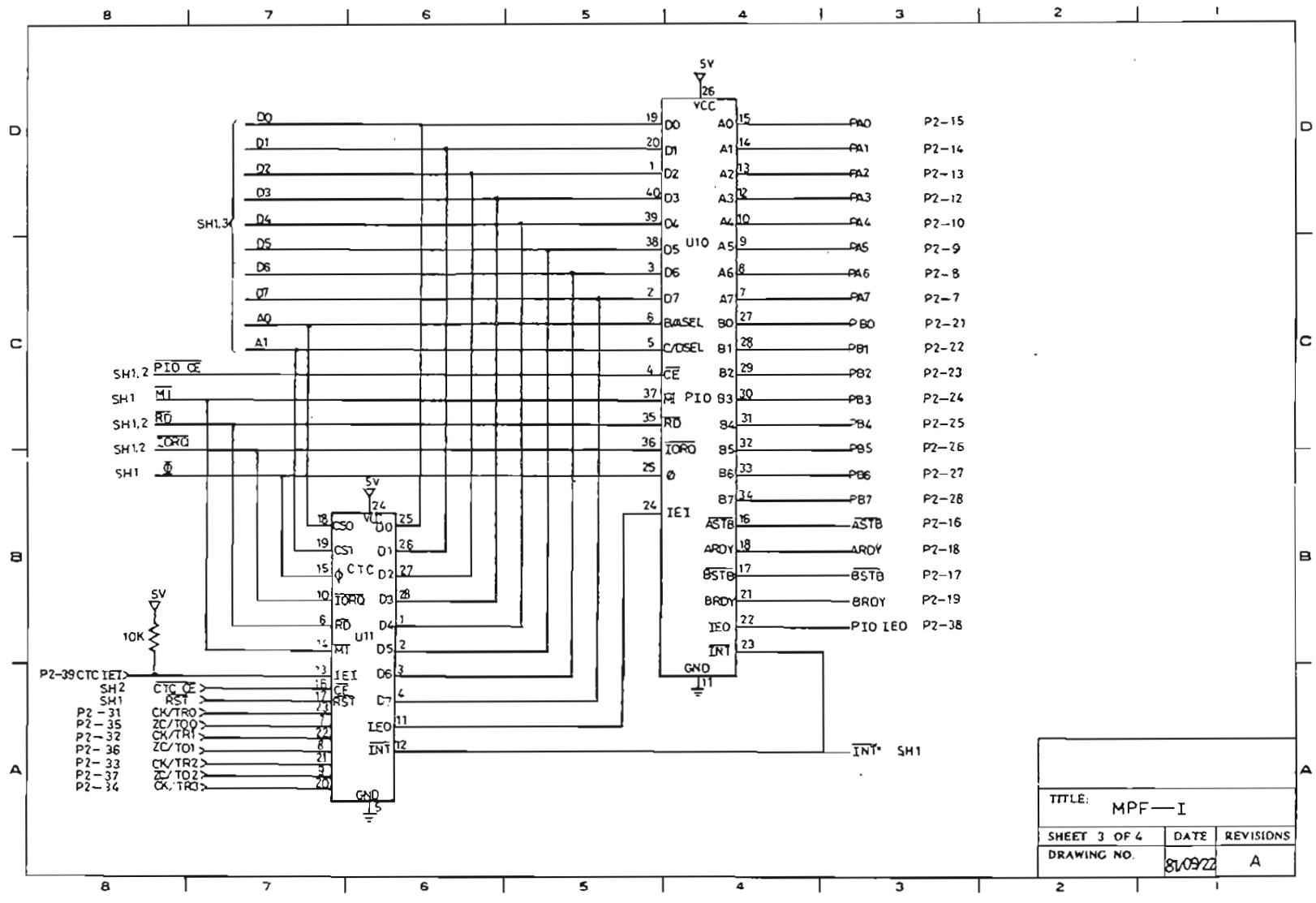
U96 (74LS139) is an I/O port decoder.

IORQ	A7	A6	Selected I/O	Port Address
0	0	0	8255	00 - 03
0	0	1	CTC	40 - 43
0	1	0	PIO	80 - 83

Note; I/O port is not fully decoded, e.g. the 16 combinations 00 - 03, 04 - 07, 08 - 0B,.....3C - 3F, all select the s 8255. The CTC & PIO are also selected by 16 different combinations.


TITLE: MPF—I
SHEET 1 OF 4
DRAWING NO.
DATE
8/09/22
REVISIONS
A
Z80 CPU
74LS14
74LS90
74LS74
74LS138
74HCT74
2716 Or 2732 Or 2764 Or 8264
2N3904
2N3906
3.58M
10PF
1K
10K
4.7UF
330
MON1
RST
INTR
HALT
BREAK


RA3
+5V
2N2222A OR 2N3904 X8
RA2
10K
SW13
D0
D1
D2
D3
D4
D5
D6
D7
U14
8255
SW1 RST
RD
WR
A0
A1
SW1,3
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
U13
75492
38 cathodes
38 cathodes
TONE
330
R8
2N9015
Q2
R9
JUMPER 2
SW1 BREAK
10K
RA1
.001UF
C13
R11 10K
R12 10K
.001UF
C12
MIC
330
R13
74LS139
U9b
SW1 A6
SW1 A7
SW1,3 IORQ
CTC CE SW3
PIO CE SW3
470PF C7
USER
74LS14
U3
CR2
1N914
330
R14
EAR
.001UF
C11
1N914
CR1
TITLE.
MPF — I
SHEET 2 OF 4
DRAWING NO.
DATE
81/09/24
REVISIONS
A

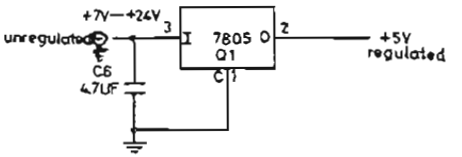

5V
VCC
U10
PIO
U11
CTC
GND
SH1,3
SH1,2
SH1
10K
P2-39 CTC IEI
SH2
P2-31
P2-35
P2-32
P2-36
P2-33
P2-37
P2-34
PIO IEO
INT* SH1
TITLE: MPF—I
SHEET 3 OF 4
DATE
REVISIONS
DRAWING NO.
81/09/22
A

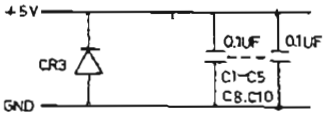
P1 PIN FUNCTION

PIN NO	SIGNAL	PIN NO	SIGNAL
1	A11	21	A10
2	A12	22	A9
3	A13	23	A8
4	A14	24	A7
5	A15	25	A6
6	CLK	26	A5
7	D4	27	A4
8	D3	28	A3
9	D5	29	A2
10	D6	30	A1
11	NC	31	A0
12	D2	32	GND
13	D7	33	$\overline{RFSH}$
14	D0	34	$\overline{M1}$
15	D1	35	$\overline{RESET}$
16	$\overline{INT}$	36	$\overline{BUSRQ}$
17	$\overline{NMI}$	37	$\overline{WAIT}$
18	$\overline{HALT}$	38	$\overline{BUSAK}$
19	$\overline{MREQ}$	39	$\overline{WR}$
20	$\overline{IORQ}$	40	$\overline{RD}$

P2 PIN FUNCTION

PIN NO	SIGNAL	PIN NO	SIGNAL
1	NC	21	PB0
2	NC	22	PB1
3	NC	23	PB2
4	NC	24	PB3
5	NC	25	PB4
6	NC	26	PB5
7	PA7	27	PB6
8	PA6	28	PB7
9	PA5	29	NC
10	PA4	30	GND
11	GND	31	CK/TR0
12	PA3	32	CK/TR1
13	PA2	33	CK/TR2
14	PA1	34	CK/TR3
15	PA0	35	ZC/TO0
16	$\overline{ASTB}$	36	ZC/TO1
17	$\overline{BSTB}$	37	ZC/TO2
18	ARDY	38	PIO IEO
19	BRDY	39	CTC IEI
20	NC	40	NC





NOTE: 1 U7 is optional, it may be 2716, 2732, 2764, 6264

2 If you want to expand the memory size of U6 to 6K, Qa(2N3904), Ra(1K), Rb(1K), Rb(4.7K), Cb(100p), must be added.

JP3 ABC	JP4 DEF	JP5 GHI	U6 2764	U7 SEE NOTE
o o—o	o—o o	o o—o	4K (0 ~ 0FFFH)	4K (2000 ~ 2FFFH)
o o—o	o o—o	o—o o	4K (0 ~ 0FFFH)	8K (2000 ~ 3FFFH)
o—o o	o o—o	o—o o	6k (0 ~ 17FFH)	8K (2000 ~ 3FFFH)

TITLE: PF—I		
SHEET 4 OF 4	DATE	REVISIONS
DRAWING NO.	81/09/22	A

E. Matrix Keyboard and Display

U14 (8255) has 3 I/O ports, PB0-PB7 control individual segments in a display, U15 and U12 are segment drivers, PC0-PC5 select which display is to be activated, U13 (75492) is a 6-digit digit driver.

The LED display uses a Multiplexing method, only one is selected at a time, from right to left. Due to its rapid multiplexing rate. The displays appear to be on continuously.

Whenever the diaplays are accesed keyboard activity is also checked via U14 (8255) PA0-PA5. If no key is pressed, PA0-PA5 are high, when there is one key pressed, via keyboard scan routine the CPU will detect which key id pressed. In MPF-I there are 6 x 6 = 36 keys, but only 32 keys are checked through the key board matrix.

F. User-Key

The user-key is not assigned a function and is reserved for user's future use. The state of this key is detected via PA6 of 8255. via PA6 of 8255.

G. Audio Tape Interface.

The program or data to be stored in Magnetic Tape is serially sent out via PC7 of 8255. The filtering & decaying circuit are composed of C13, R11, C12, R12 and R13. This decayed signal is to MIC ("Microphone") inlet of Tape-recorder, Q2 drives an LED and speaker. PC7 is also used as the port for audio output.

A recorded file may be read back to the RAM from the ("Earphone") EAR outlet of Tape recorder. The input interface circuit is composed of R14, CR2, CR1 and C11. This circuit converts EAR inputed signal to TTL level signal and detected by CPU via PA7 of 8255.

H. Step, Break point and Monitor Break

PC6 is normally high. This signal send to R0 input of U4 (7490) will preset U4 output to 0000, and make NMI of Z-80 high. When PC reach Breakpoint or MPF-I execute single step, PC6 will output low, U4 starts counting, after 5th OP code fetch, NMI becomes low. This will interrupt program execution and jump back to monitor program.

Logic State of U4 (74LS90)

	R9	R0	Qa	Qd	Qc	Qb	NMI	Comment
Normal State	0	1	0	0	0	0	1	U4 preset to 0000
BREAK becomes low	0	0	0	0	0	0	1	R0=$\overline{\text{BREAK}}$=0
1St MI	0	0	0	0	0	1	1	7490 Start commting
2nd MI	0	0	0	0	1	0	1	Qd,Qc,Qb is Mod, 5
3rd MI	0	0	0	0	1	1	1	Counter
4th MI	0	0	0	1	0	0	1	
5th MI	0	0	1	0	0	0	0	Qa from 0-1 when Qd from 1-0
Pressing Key MONI	1	0	1	1	0	0	0	U4 Preset to 1001

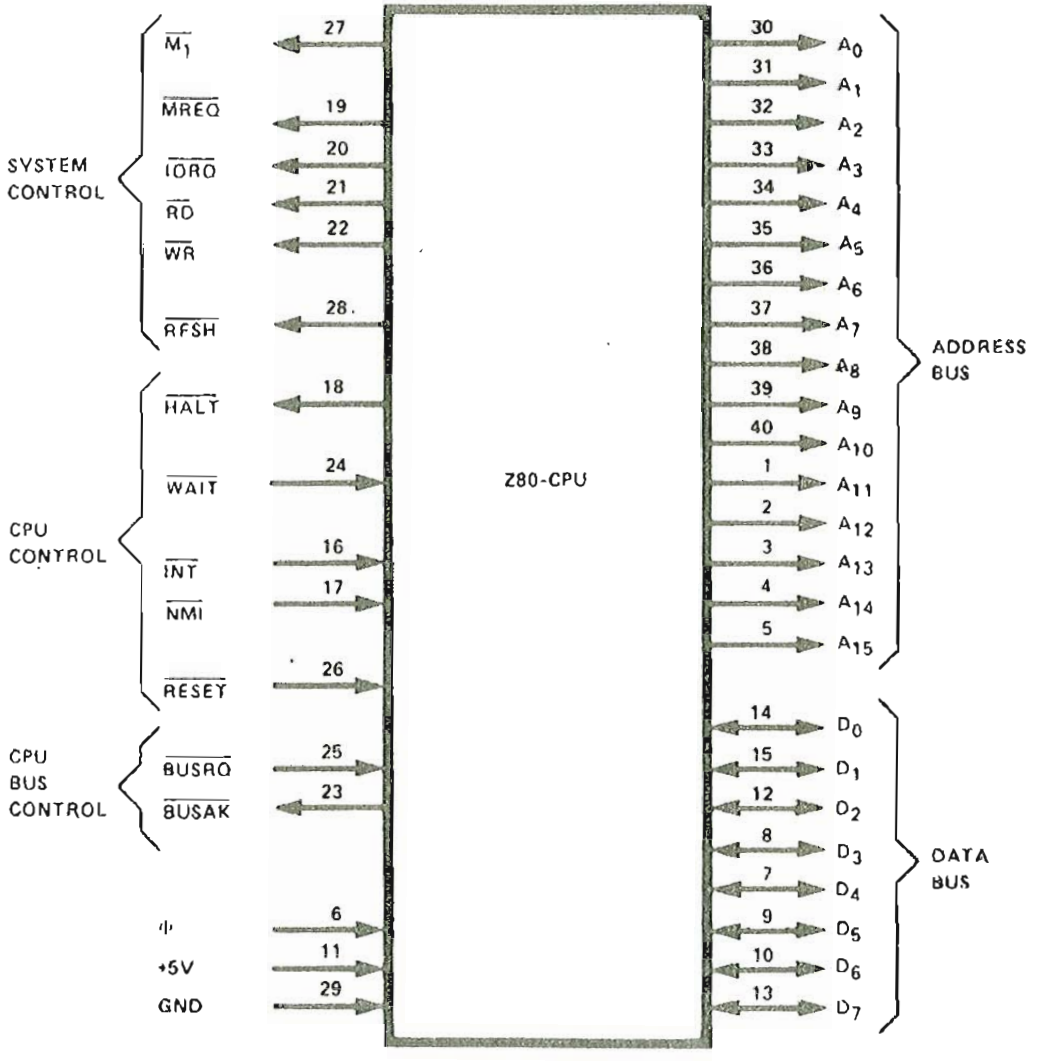
After MONI key is pressed, R9 of U4 is high, Qa becomes high and $\overline{\text{NMI}}$ becomes Low. So CPU jump back to monitor program execution due to nonmaskable interrupt.

I. PIO and CTC

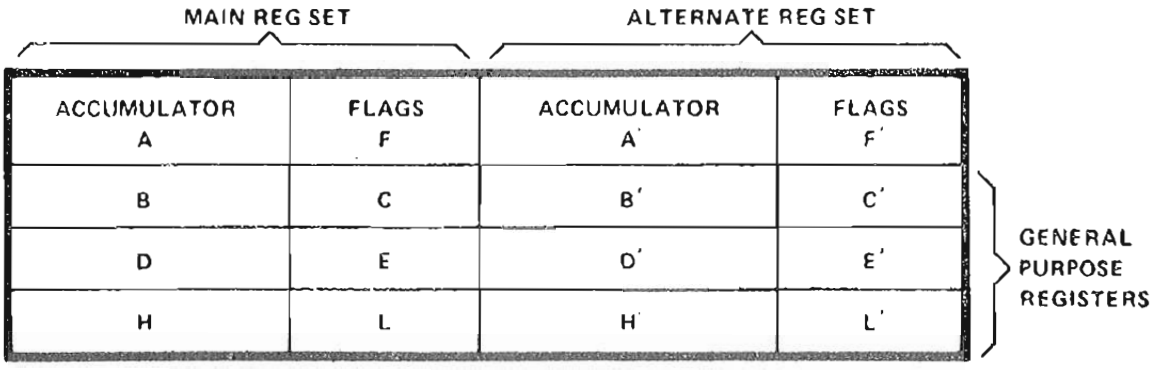
U11 (CTC) and U10 (PIO) are daisy-chained, CTC has the higher interrupt priority, CTC IEI, PIO IEO, CTC channel signals and PIO I/O port are reserved on P2 edge connector for user future expansion.

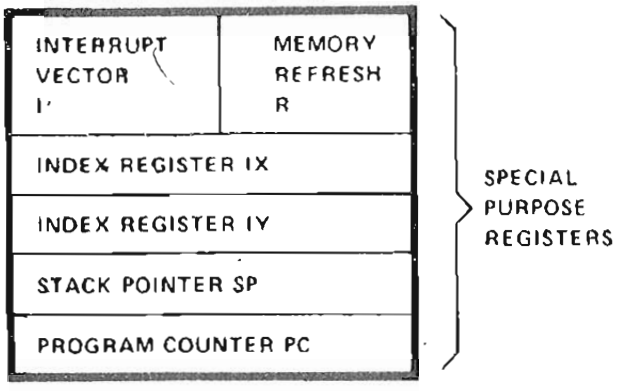
APPENDIX C

Z80-CPU Programming reference.


SYSTEM CONTROL
CPU CONTROL
CPU BUS CONTROL
M1
MREQ
IORQ
RD
WR
RFSH
HALT
WAIT
INT
NMI
RESET
BUSRQ
BUSAK
φ
+5V
GND
27
19
20
21
22
28
18
24
16
17
26
25
23
6
11
29
Z80-CPU
30
31
32
33
34
35
36
37
38
39
40
1
2
3
4
5
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
ADDRESS BUS
14
15
12
8
7
9
10
13
D0
D1
D2
D3
D4
D5
D6
D7
DATA BUS

CPU PIN-OUTS





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	1	X	P	0	0	Logical operations
OR s; XOR s	↕	↕	X	0	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'

SOURCE

DESTINATION			IMPLIED		REGISTER							REG INDIRECT			INDEXED		EXT ADDR	IMME
			I	R	A	B	C	D	E	H	L	(HL)	(BC)	(DE)	(IX+d)	(IY+d)	(nn)	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													

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 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
		BC								01 n n	ED 4B n n	C1
		DE								11 n n	ED 5B n n	D1
		HL								21 n n	2A n n	E1
		SP				F9		DD F9	FD F9	31 n n	ED 7B n n	
		IX								DD 21 n n	DD 2A n n	DD E1
		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			
PUSH INSTRUCTIONS	REG. IND.	(SP)	F5	C5	D5	E5		DD E5	FD E5			

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

POP INSTRUCTIONS

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			E8		
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		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		2	2	7	000 B
										← n →					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	DD	3	5	19	100 H
										10 000 110					101 L
										← d →					111 A
ADD A, (IY+d)	A ← A+(IY+d)	↕	↕	X	↕	X	V	0	↕	11 111 101	FD	3	5	19	
										10 000 110					
										← d →					
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	0	X	P	0	0	110					The indicated bits
XOR s	A ← A ⊕ s	↕	↕	X	0	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	DD	3	6	23	
										00 110 100					
										← d →					
INC (IY+d)	(IY+d) ← (IY+d)+1	↕	↕	X	↕	X	V	0	•	11 111 101	FD	3	6	23	
										00 110 100					
										← d →					
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 0038H
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	$A \leftarrow \overline{A}$	•	•	X	1	X	•	1	•	00 101 111	2F	1	1	4	Complement accumulator (One's complement)
NEG	$A \leftarrow \overline{A} + 1$	↕	↕	X	↕	X	V	1	↕	11 101 101 01 000 100	ED 44	2	2	8	Negate acc. (two's complement)
CCF	$CY \leftarrow \overline{CY}$	•	•	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	Flags								Op-Code		No. of Bytes	No. of M Cycles	No. of T States	Comments	
		S	Z		H		P/V	N	C	76 543 210	Hex					
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	ED	2	4	15	01	DE
										01 ss1 010					10	HL
															11	SP
SBC HL, ss	HL ← HL-ss-CY	↕	↕	X	X	X	V	1	↕	11 101 101	ED	2	4	15		
										01 ss0 010						
ADD IX, pp	IX ← IX + pp	•	•	X	X	X	•	0	↕	11 011 101	DD	2	4	15	pp	Reg
										00 pp1 001					00	BC
															01	DE
															10	IX
															11	SP
ADD IY, rr	IY ← IY + rr	•	•	X	X	X	•	0	↕	11 111 101	FD	2	4	15	rr	Reg
										00 rr1 001					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	DD	2	2	10		
										00 100 011	23					
INC IY	IY ← IY + 1	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	10		
										00 100 011	23					
DEC ss	ss ← ss - 1	•	•	X	•	X	•	•	•	00 ss1 011		1	1	6		
DEC IX	IX ← IX - 1	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	10		
										00 101 011	2B					
DEC IY	IY ← IY - 1	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	10		
										00 101 011	2B					

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

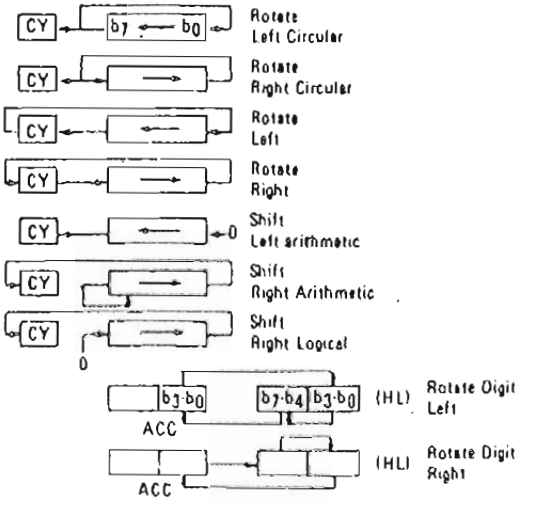
Flag Notation: • = flag not affected, 0 = flag reset, 1 = flag set, X = flag is unknown.
↕ = flag is affected according to the result of the operation.

ROTATES AND SHIFTS

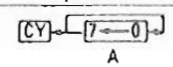
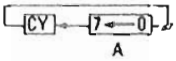
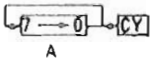
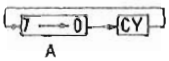
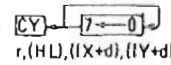
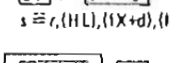
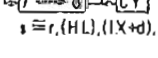
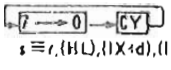
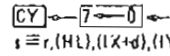
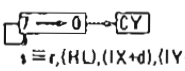
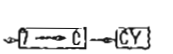
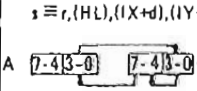
Source and Destination

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		•	•	X	0	X	•	0	‡	00 000 111	07	1	1	4	Rotate left circular accumulator
RLA		•	•	X	0	X	•	0	‡	00 010 111	17	1	1	4	Rotate left accumulator
RRCA		•	•	X	0	X	•	0	‡	00 001 111	0F	1	1	4	Rotate right circular accumulator
RRA		•	•	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)	 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	 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	 s ≡ r,(HL),(IX+d),(IY+d)	‡	‡	X	0	X	P	0	‡	001					
RR s	 s ≡ r,(HL),(IX+d),(IY+d)	‡	‡	X	0	X	P	0	‡	011					
SLA s	 s ≡ r,(HL),(IX+d),(IY+d)	‡	‡	X	0	X	P	0	‡	100					
SRA s	 s ≡ r,(HL),(IX+d),(IY+d)	‡	‡	X	0	X	P	0	‡	101					
SRL s	 s ≡ r,(HL),(IX+d),(IY+d)	‡	‡	X	0	X	P	0	‡	111					
RLD		‡	‡	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). The content of the upper half of the accumulator is unaffected
RRD		‡	‡	X	0	X	P	0	•	11 101 101 01 100 111	ED 67	2	5	18	

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 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
BIT b, r	$Z \leftarrow \overline{r_b}$	X	↕	X	1	X	X	0	•	11 001 011	CB	2	2	8	r / Reg.: 000 B, 001 C, 010 D, 011 E, 100 H, 101 L, 111 A
										01 b r					
BIT b, (HL)	$Z \leftarrow \overline{(HL)_b}$	X	↕	X	1	X	X	0	•	11 001 011	CB	2	3	12	
										01 b 110					
BIT b, (IX+d)	$Z \leftarrow \overline{(IX+d)_b}$	X	↕	X	1	X	X	0	•	11 011 101	DD	4	5	20	
										11 001 011	CB				
										- d -					
										01 b 110					
BIT b, (IY+d)	$Z \leftarrow \overline{(IY+d)_b}$	X	↕	X	1	X	X	0	•	11 111 101	FD	4	5	20	b / Bit Tested: 000 0, 001 1, 010 2, 011 3, 100 4, 101 5, 110 6, 111 7
										11 001 011	CB				
										- d -					
										01 b 110					
SET b, r	$r_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011	CB	2	2	8	
										11 b r					
SET b, (HL)	$(HL)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011	CB	2	4	15	
										11 b 110					
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 011 101	DD	4	6	23	
										11 001 011	CB				
										- d -					
										11 b 110					
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 111 101	FD	4	6	23	
										11 001 011	CB				
										- d -					
										11 b 110					
RES b, s	$s_b \leftarrow 0$ $s \equiv r, (HL), (IX+d), (IY+d)$									10					To form new O_p Code replace 11 of SET b, s with 10. Flags and time states for SET instruction

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 'JP'	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	PARITY ODD	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		H		P/V	N	C	Op-Code 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
CALL nn	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ $PC \leftarrow nn$	•	•	X	•	X	•	•	•	11 001 101 - n - - n -	CD	3	5	17	
CALL cc, nn	If condition cc is false continue, otherwise same as CALL nn	•	•	X	•	X	•	•	•	11 cc 100 - n - - n -		3	3	10	If cc is false If cc is true
RET	$PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP+1)$	•	•	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	5	If cc is false If cc is true
RETI	Return from interrupt	•	•	X	•	X	•	•	•	11 101 101 01 001 101	ED 4D	2	4	14	
RETN[1]	Return from non maskable interrupt	•	•	X	•	X	•	•	•	11 101 101 01 000 101	ED 45	2	4	14	
RST p	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ $PC_H \leftarrow 0$ $PC_L \leftarrow p$	•	•	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

			PORT ADDRESS	
			IMMED.	REG. INDIR.
			n	(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

INPUT DESTINATION

BLOCK INPUT COMMANDS

OUTPUT GROUP

			SOURCE REGISTER							REG. IND.
			A	B	C	D	E	H	L	(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 $A_8 \sim A_{15}$

Notes: (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 INSTRUCTIONS SORTED BY OP-CODE

OBJ CODE	SOURCE STATEMENT
00	NOP
018405	LD BC,NN
02	LD (BC),A
03	INC BC
04	INC B
05	DEC B
0620	LD B,N
07	RLCA
08	EX AF,AF'
09	ADD HL,BC
0A	LD A,(BC)
0B	DEC BC
0C	INC C
0D	DEC C
0E20	LD C,N
0F	RRCA
102E	DJNZ DIS
118405	LD DE,NN
12	LD (DE),A
13	INC DE
14	INC D
15	DEC D
1620	LD D,N
17	RLA
182E	JR DIS
19	ADD HL,DE
1A	LD A,(DE)
1B	DEC DE
1C	INC E
1D	DEC E
1E20	LD E,N
1F	RRA
202E	JR NZ,DIS
218405	LD HL,NN
228405	LD (NN),HL
23	INC HL
24	INC H
25	DEC H
2620	LD H,N
27	DAA
282E	JR Z,DIS
29	ADD HL,HL
2A8405	LD HL,(NN)
2B	DEC HL
2C	INC L
2D	DEC L
2E20	LD L,N
2F	CPL
302E	JR NC,DIS
318405	LD SP,NN
328405	LD (NN),A
33	INC SP
34	INC (HL)
35	DEC (HL)
3620	LD (HL),N
37	SCF
382E	JR C,DIS
39	ADD HL,SP
3A8405	LD A,(NN)
3B	DEC SP
3C	INC A
3D	DEC A
3E20	LD A,N
3F	CCF
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
46	LD B,(HL)
47	LD B,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
4E	LD C,(HL)
4F	LD C,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
56	LD D,(HL)
57	LD D,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
5E	LD E,(HL)
5F	LD E,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
66	LD H,(HL)
67	LD H,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
6E	LD L,(HL)
6F	LD L,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
76	HALT
77	LD (HL),A
78	LD A,B
79	LD A,C
7A	LD A,D
7B	LD A,E
7C	LD A,H
7D	LD A,L
7E	LD A,(HL)
7F	LD 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
86	ADD A,(HL)
87	ADD 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
8E	ADC A,(HL)
8F	ADC A,A
90	SUB B
91	SUB C
92	SUB D
93	SUB E
94	SUB H
95	SUB L
96	SUB (HL)
97	SUB 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
9E	SBC A,(HL)
9F	SBC A,A
A0	AND B
A1	AND C
A2	AND D
A3	AND E
A4	AND H
A5	AND L
A6	AND (HL)
A7	AND A
A8	XOR B
A9	XOR C
AA	XOR D
AB	XOR E
AC	XOR H
AD	XOR L
AE	XOR (HL)
AF	XOR A
B0	OR B
B1	OR C
B2	OR D
B3	OR E
B4	OR H
B5	OR L
B6	OR (HL)
B7	OR A

B8	CP B
B9	CP C
BA	CP D
BB	CP E
BC	CP H
BD	CP L
BE	CP (HL)
BF	CP A
C0	RET NZ
C1	POP BC
C28405	JP NZ,NN
C38405	JP NN
C48405	CALL NZ,NN
C5	PUSH BC
C620	ADD A,N
C7	RST O
C8	RET Z
C9	RET
CA8405	JP Z,NN
CC8405	CALL Z,NN
CD8405	CALL NN
CE20	ADC A,N
CF	RST 8
D0	RET NC
D1	POP DE
D28405	JP NC,NN
D320	OUT (N),A
D48405	CALL NC,NN
D5	PUSH DE
D620	SUB N
D7	RST 10H
D8	RET C
D9	EXX
DA8405	JP C,NN
DB20	IN A,(N)
DC8405	CALL C,NN
DE20	SBC A,N
DF	RST 18H
E0	RET PO
E1	POP HL
E28405	JP PO,NN
E3	EX (SP),HL
E48405	CALL PO,NN
E5	PUSH HL
E620	AND N
E7	RST 20H
E8	RET PE
E9	JP (HL)

EA8405	JP PE,NN
EB	EX DE,HL
EC8405	CALL PE,NN
EE20	XOR N
EF	RST 28H
F0	RET P
F1	POP AF
F28405	JP P,NN
F3	DI
F48405	CALL P,NN
F5	PUSH AF
F620	OR N
F7	RST 30H
F8	RET M
F9	LD SP,HL
FA8405	JP M,NN
FB	EI
FC8405	CALL M,NN
FE20	CP N
FF	RST 38H
CB00	RLC B
CB01	RLC C
CB02	RLC D
CB03	RLC E
CB04	RLC H
CB05	RLC L
CB06	RLC (HL)
CB07	RLC A
CB08	RRC B
CB09	RRC C
CB0A	RRC D
CB0B	RRC E
CB0C	RRC H
CB0D	RRC L
CB0E	RRC (HL)
CB0F	RRC A
CB10	RL B
CB11	RL C
CB12	RL D
CB13	RL E
CB14	RL H
CB15	RL L
CB16	RL (HL)
CB17	RL A
CB18	RR B
CB19	RR C
CB1A	RR D
CB1B	RR E

CB1C	RR H
CB1D	RR L
CB1E	RR (HL)
CB1F	RR A
CB20	SLA B
CB21	SLA C
CB22	SLA D
CB23	SLA E
CB24	SLA H
CB25	SLA L
CB26	SLA (HL)
CB27	SLA A
CB28	SRA B
CB29	SRA C
CB2A	SRA D
CB2B	SRA E
CB2C	SRA H
CB2D	SRA L
CB2E	SRA (HL)
CB2F	SRA A
CB38	SRL B
CB39	SRL C
CB3A	SRL D
CB3B	SRL E
CB3C	SRL H
CB3D	SRL L
CB3E	SRL (HL)
CB3F	SRL 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
CB46	BIT 0,(HL)
CB47	BIT 0,A
CB48	BIT 1,B
CB49	BIT 1,C
CB4A	BIT 1,D
CB4B	BIT 1,E
CB4C	BIT 1,H
CB4D	BIT 1,L
CB4E	BIT 1,(HL)
CB4F	BIT 1,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
CB56	BIT 2,(HL)
CB57	BIT 2,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
CB5E	BIT 3,(HL)
CB5F	BIT 3,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
CB66	BIT 4,(HL)
CB67	BIT 4,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
CB6E	BIT 5,(HL)
CB6F	BIT 5,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
CB76	BIT 6,(HL)
CB77	BIT 6,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
CB7E	BIT 7,(HL)
CB7F	BIT 7,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
CB86	RES 0,(HL)
CB87	RES 0,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
CB8E	RES 1,(HL)
CB8F	RES 1,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
CB96	RES 2,(HL)
CB97	RES 2,A
CB98	RES 3,B
CB99	RES 3,C
CB9A	RES 3,D
CB9B	RES 3,E
CB9C	RES 3,H
CB9D	RES 3,L
CB9E	RES 3,(HL)
CB9F	RES 3,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
CBA6	RES 4,(HL)
CBA7	RES 4,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
CBAE	RES 5,(HL)
CBAF	RES 5,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
CBB6	RES 6,(HL)
CBB7	RES 6,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
CBBE	RES 7,(HL)
CBBF	RES 7,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
CBC6	SET 0,(HL)
CBC7	SET 0,A
CBC8	SET 1,B
CBC9	SET 1,C
CBCA	SET 1,D
CBCB	SET 1,E
CBCC	SET 1,H
CBCD	SET 1,L
CBCE	SET 1,(HL)
CBCF	SET 1,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
CBD6	SET 2,(HL)
CBD7	SET 2,A
CBD8	SET 3,B
CBD9	SET 3,C
CBDA	SET 3,D
CBDB	SET 3,E
CBDC	SET 3,H
CBDD	SET 3,L
CBDE	SET 3,(HL)
CBDF	SET 3,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
CBE6	SET 4,(HL)
CBE7	SET 4,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
CBEE	SET 5,(HL)
CBEF	SET 5,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
CBF6	SET 6,(HL)
CBF7	SET 6,A
CBF8	SET 7,B
GBF9	SET 7,C
CBFA	SET 7,D
CBFB	SET 7,E
CBFC	SET 7,H
CBFD	SET 7,L
CBFE	SET 7,(HL)
CBFF	SET 7,A
DD09	ADD IX,BC
DD19	ADD IX,DE
DD218405	LD IX,NN
DD228405	LD (NN),IX
DD23	INC IX
DD29	ADD IX,IX
DD2A8405	LD IX, (NN)
DD2B	DEC IX
DD3405	INC (IX+d)
DD3505	DEC (IX+d)
DD360520	LD (IX+d),N
DD39	ADD IX,SP
DD4605	LD B,(IX+d)
DD4E05	LD C,(IX+d)
DD5605	LD D,(IX+d)
DD5E05	LD E,(IX+d)
DD6605	LD H,(IX+d)
DD6E05	LD L,(IX+d)
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
DD7705	LD (IX+d),A
DD7E05	LD A,(IX+d)
DD8605	ADD A,(IX+d)
DD8E05	ADC A,(IX+d)
DD9605	SUB (IX+d)
DD9E05	SBC A,(IX+d)
DDA605	AND (IX+d)
DDAE05	XOR (IX+d)
DDB605	OR (IX+d)
DDBE05	CP (IX+d)
DDE1	POP IX
DDE3	EX (SP),IX
DDE5	PUSH IX
DDE9	JP (IX)
DDF9	LD SP,IX
DDCB0506	RLC (IX+d)
DDCB050E	RRC (IX+d)
DDCB0516	RL (IX+d)
DDCB051E	RR (IX+d)
DDCB0526	SLA (IX+d)
DDCB052E	SRA (IX+d)
DDCB053E	SRL (IX+d)
DDCB0546	BIT 0,(IX+d)
DDCB054E	BIT 1,(IX+d)
DDCB0556	BIT 2,(IX+d)
DDCB055E	BIT 3,(IX+d)
DDCB0566	BIT 4,(IX+d)
DDCB056E	BIT 5,(IX+d)
DDCB0576	BIT 6,(IX+d)
DDCB057E	BIT 7,(IX+d)
DDCB0586	RES 0,(IX+d)
DDCB058E	RES 1,(IX+d)
DDCB0596	RES 2,(IX+d)
DDCB059E	RES 3,(IX+d)
DDCB05A6	RES 4,(IX+d)
DDCB05AE	RES 5,(IX+d)
DDCB05B6	RES 6,(IX+d)
DDCB05BE	RES 7,(IX+d)
DDCB05C6	SET 0,(IX+d)
DDCB05CE	SET 1,(IX+d)
DDCB05D6	SET 2,(IX+d)
DDCB05DE	SET 3,(IX+d)
DDCB05E6	SET 4,(IX+d)
DDCB05EE	SET 5,(IX+d)

DDCB05F6	SET 6,(IX+d)
DDCB05FE	SET 7,(IX+d)
ED40	IN B,(C)
ED41	OUT (C),B
ED42	SBC HL,BC
ED438405	LD (NN),BC
ED44	NEG
ED45	RETN
ED46	IM 0
ED47	LD I,A
ED48	IN C,(C)
ED49	OUT (C),C
ED4A	ADC HL,BC
ED4B8405	LD BC,(NN)
ED4D	RETI
ED50	IN D,(C)
ED51	OUT (C),D
ED52	SBC HL,DE
ED538405	LD (NN),DE
ED56	IM 1
ED57	LD A,I
ED58	IN E,(C)
ED59	OUT (C),E
ED5A	ADC HL,DE
ED5B8405	LD DE,(NN)
ED5E	IM 2
ED60	IN H,(C)
ED61	OUT (C),H
ED62	SBC HL,HL
ED67	RRD
ED68	IN L,(C)
ED69	OUT (C),L
ED6A	ADC HL,HL
ED6F	RLD
ED72	SBC HL,SP
ED738405	LD (NN),SP
ED78	IN A,(C)
ED79	OUT (C),A
ED7A	ADC HL,SP
ED7B8405	LD SP,(NN)
EDA0	LDI
EDA1	CPI
EDA2	INI
EDA3	OUTI
EDA8	LDD
EDA9	CPD
EDAA	IND
EDAB	OUTD

EDB0	LDIR
EDB1	CPIR
EDB2	INIR
EDB3	OTIR
EDB8	LDDR
EDB9	CPDR
EDBA	INDR
EDBB	OTDR
FD09	ADD IY,BC
FD19	ADD IY,DE
FD218405	LD IY,NN
FD228405	LD (NN),IY
FD23	INC IY
FD29	ADD IY,IY
FD2A8405	LD IY,(NN)
FD2B	DEC IY
FD3405	INC (IY+d)
FD3505	DEC (IY+d)
FD360520	LD (IY+d),N
FD39	ADD IY,SP
FD4605	LD B,(IY+d)
FD4E05	LD C,(IY+d)
FD5605	LD D,(IY+d)
FD5E05	LD E,(IY+d)
FD6605	LD H,(IY+d)
FD6E05	LD L,(IY+d)
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
FD7705	LD (IY+d),A
FD7E05	LD A,(IY+d)
FD8605	ADD A,(IY+d)
FD8E05	ADC A,(IY+d)
FD9605	SUB (IY+d)
FD9E05	SBC A,(IY+d)
FDA605	AND (IY+d)
FDAE05	XOR (IY+d)
FDB605	OR (IY+d)
FDBE05	CP (IY+d)
FDE1	POP IY
FDE3	EX (SP),IY
FDE5	PUSH IY
FDE9	JP (IY)
FDF9	LD SP,IY
FDCB0506	RLC (IY+d)

FDCB050E	RRC (IY+d)
FDCB0516	RL (IY+d)
FDCB051E	RR (IY+d)
FDCB0526	SLA (IY+d)
FDCB052E	SRA (IY+d)
FDCB053E	SRL (IY+d)
FDCB0546	BIT 0,(IY+d)
FDCB054E	BIT 1,(IY+d)
FDCB0556	BIT 2,(IY+d)
FDCB055E	BIT 3,(IY+d)
FDCB0566	BIT 4,(IY+d)
FDCB056E	BIT 5,(IY+d)
FDCB0576	BIT 6,(IY+d)
FDCB057E	BIT 7,(IY+d)
FDCB0586	RES 0,(IY+d)
FDCB058E	RES 1,(IY+d)
FDCB0596	RES 2,(IY+d)
FDCB059E	RES 3,(IY+d)
FDCB05A6	RES 4,(IY+d)
FDCB05AE	RES 5,(IY+d)
FDCB05B6	RES 6,(IY+d)
FDCB05BE	RES 7,(IY+d)
FDCB05C6	SET 0,(IY+d)
FDCB05CE	SET 1,(IY+d)
FDCB05D6	SET 2,(IY+d)
FDCB05DE	SET 3,(IY+d)
FDCB05E6	SET 4,(IY+d)
FDCB05EE	SET 5,(IY+d)
FDCB05F6	SET 6,(IY+d)
FDCB05FE	SET 7,(IY+d)

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

BD	CP L
FE20	CP N
EDA9	CPD
ED89	CPDR
EDA1	CPI
ED81	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
102F	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
ED82	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)

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
EDB3	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

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

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
```

Z80 – CPU INTERRUPT STRUCTURE

MASKABLE ($\overline{INT}$)

Mode 0

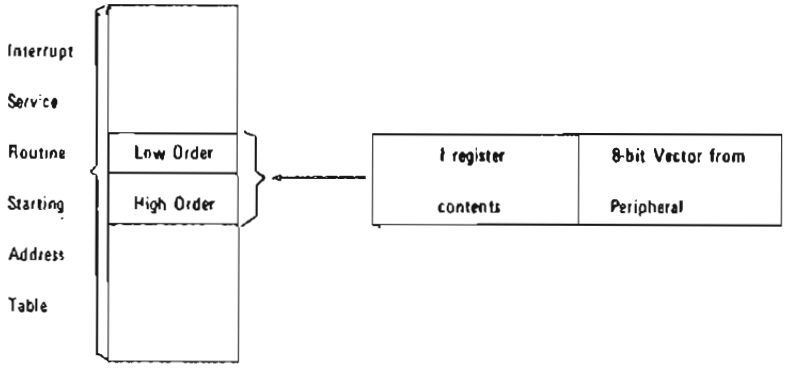
Place instruction onto Data Bus during $\overline{INTA} = \overline{M1} \bullet \overline{IORQ}$ like 8080A

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

Z80-PIO PIN ASSIGNMENT

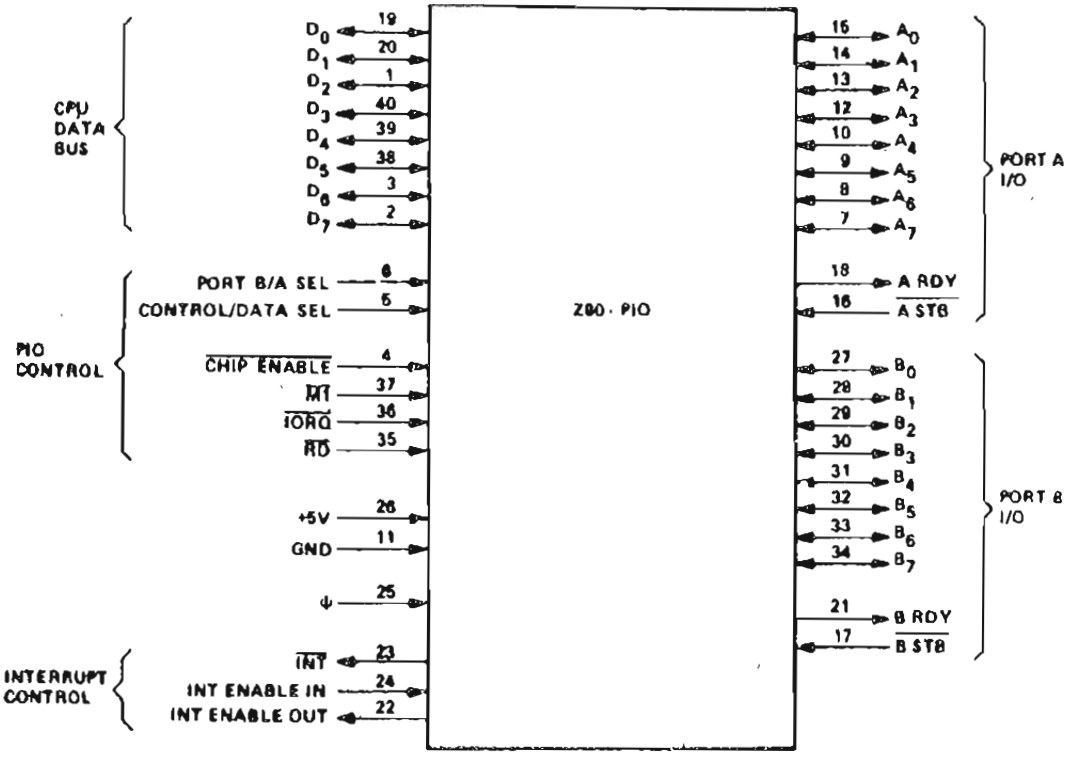


FIGURE 3.0-1
PIO PIN CONFIGURATION

REGISTER SELECTION

SELECT LINES C/D	B/A	REGISTER SELECTED
0	0	A Data
0	1	B Data
1	0	A Control
	1	B Control

LOAD INTERRUPT VECTOR

D7							D0	
V7	V6	V5	V4	V3	V2	V1	0	Control Register

SET OPERATING MODE

D7							D0	
M1	M0	X	X	1	1	1	1	Control Register

Mode Number	M1	M0	Mode
0	0	0	Output
1	0	1	Input
2	1	0	Bidirectional
3	1	1	Bit Control

If Mode 3 selected, the next control word is

D7							D0	
I/O_7	I/O_6	I/O_5	I/O_4	I/O_3	I/O_2	I/O_1	I/O_0	Control Register

I/O = 1 Sets bit to Input

I/O = 0 Sets bit to Output

SET INTERRUPT CONTROL

D7							D0	
Int Enable	AND/ OR	High/ Low	Mask Follows	0	1	1	1	Control Register

In Mode 3 if Mask follows = 1, the next control word is

D7							D0	
MB_7	MB_6	MB_5	MB_4	MB_3	MB_2	MB_1	MB_0	Control Register

MB = 0 Monitor the bit

MB = 1 Mask the bit

ENABLE / DISABLE INTERRUPTS

D7							D0	
Int Enable	X	X	X	0	0	1	1	Control Register

3.0 CTC PIN DESCRIPTION

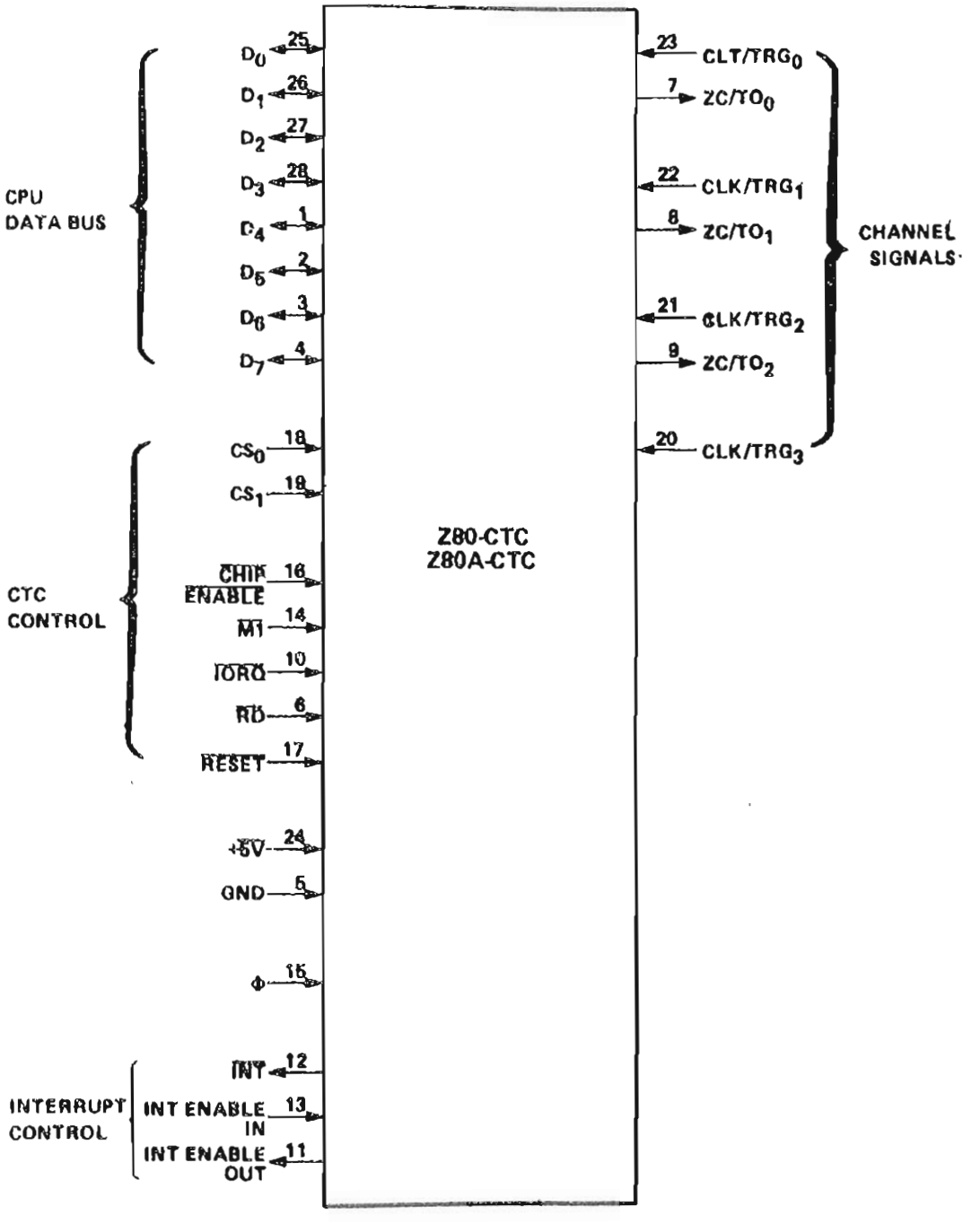


FIGURE 3.0-1
CTC PIN CONFIGURATION

CTC PROGRAMMING SUMMARY

REGISTER SELECTION

SELECT LINES CS_1	SELECT LINES CS_0	CHANNEL SELECTED	PRIORITY
0	0	0	Highest
0	1	1	
1	0	2	
1	1	3	Lowest

READ = DOWN COUNTER WRITE = CONTROL REGISTER

LOAD INTERRUPT VECTOR

$CS_0 = CS_1 = 0$

D7							D0	
V_7	V_6	V_5	V_4	V_3	X	X	0	Control Register

XX is the binary equivalent of interrupting channel number

SET OPERATING MODE

Timer Mode only

D7							D0	
Interrupt Enable	Mode	Range	Slope	Trigger	Load Time Constant	Reset	1	Control Register
	Counter/Timer	256/16	+/-	On/Off				

If Load Time Constant = 1 the next control word is the Time Constant:

D7							D0
TC_7	TC_6	TC_5	TC_4	TC_3	TC_2	TC_1	TC_0

CTC Channel interrupts when 01_H is decremented to 00_H

Time Content	Decimal counts to interrupt
01_H	1
.	.
.	.
FF_H	255
00_H	256

APPENDIX D

Reference Books:

1. Z80 Assembly Language programming Manual.
2. Intel Component Data Catalog.
3. The TTL handbook.

PART NO.: 49.00710.001