Micro-Professor™

MPF-II User's Manual

MULTITECH INDUSTRIAL CORPORATION

Micro-Professor™

MPF-II User's Manual

CHAPTER 1 UMPACKING AND INSTALLATION

CHAPTER 2 HARDWARE DESCRIPTION

CHAPTER 3 MONITOR PROGRAM

CHAPTER 4 MEMORY STRUCTURE

CHAPTER 5 INPUT/OUTPUT

CHAPTER 1

UNPACKING AND INSTALLATION

1 • 1 Unpacking

Welcome to use the MPF-II! At first, please check the following parts you should have with your MPF-II:

- MPF-II Microcomputer (Fig.1-1)
- AC-DC Switching Power Supply (Fig.1-2)
- TV/Video Monitor Interfacing Cable (Fig.1-3)
- TV/Computer Switch Box (Fig.1-4)
- Recorder Line
- "Micro-Nurse"--Self-Diagnosis Cassette
- Two Name-Plates--"One Key Basic" and "Graphic"
- Manual for "Micro-Nurse"
- Installation Manual
- MPF-II User's Manual

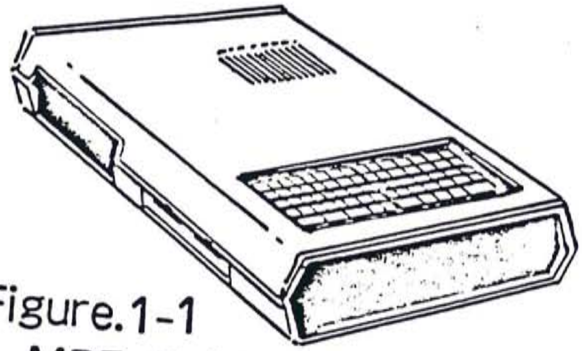


Figure.1-1
MPF-II Microcomputer

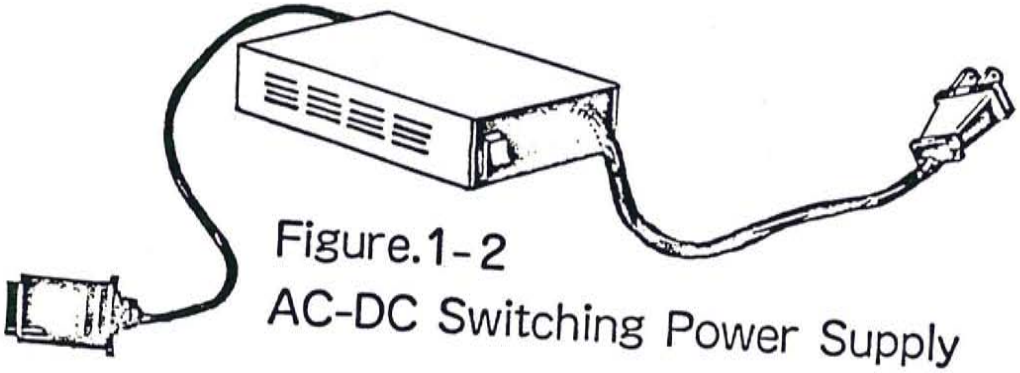
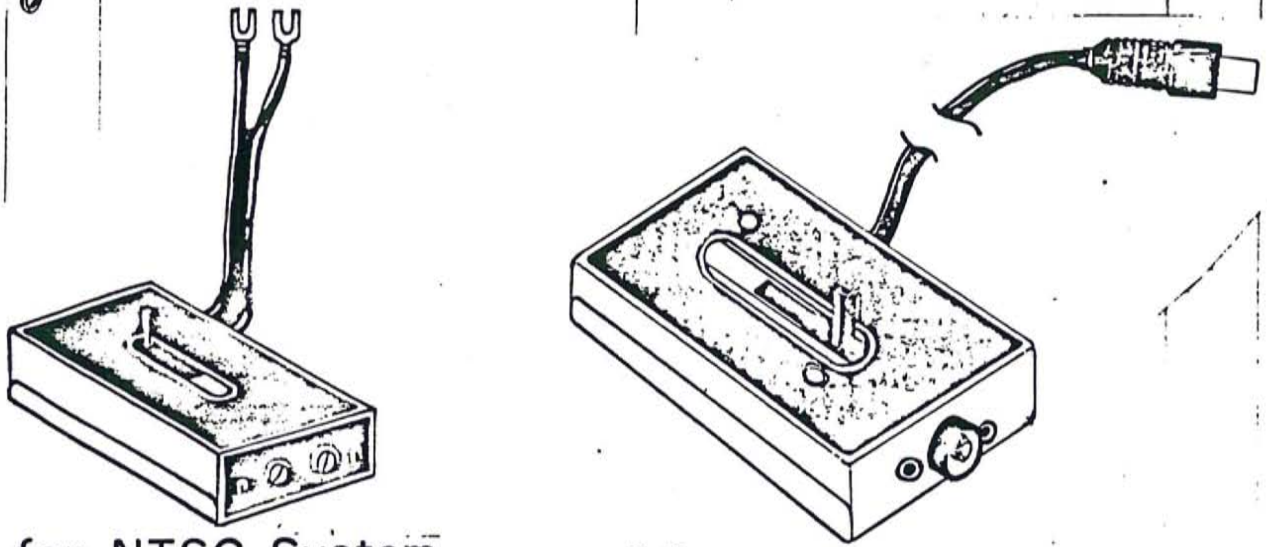


Figure.1-2
AC-DC Switching Power Supply



for NTSC System

for PAL System

Figure.1-4 Switch Box

1 • 2 Connecting TV

1. Connect the cable from the MPF-II to TV/Computer Swich Box. (Fig.1-5)
2. Disconnect the outdoor VHF antenna from the television set, connect it to TV/Computer Switch Box which is switched to the side marked "TV". (Fig.1-5)
3. Connect the two cables from the TV/Computer Switch Box to VHF antenna terminals on the television set. (Fig.1-5)
4. Switch the TV/Computer Switch Box to the side marked "COMPUTER". If you want to watch TV, just slide the switch to the "TV" side.
5. If you have a coaxial cable, disconnect the cable from your television, then screw the cable into the impedance-matching transformer and attach it to the switching box. (Consult your local Multitech dealer or distributor, if you don't know exactly what to do.)
6. If you have a monitor, connect the cable from MPF-II to the monitor.

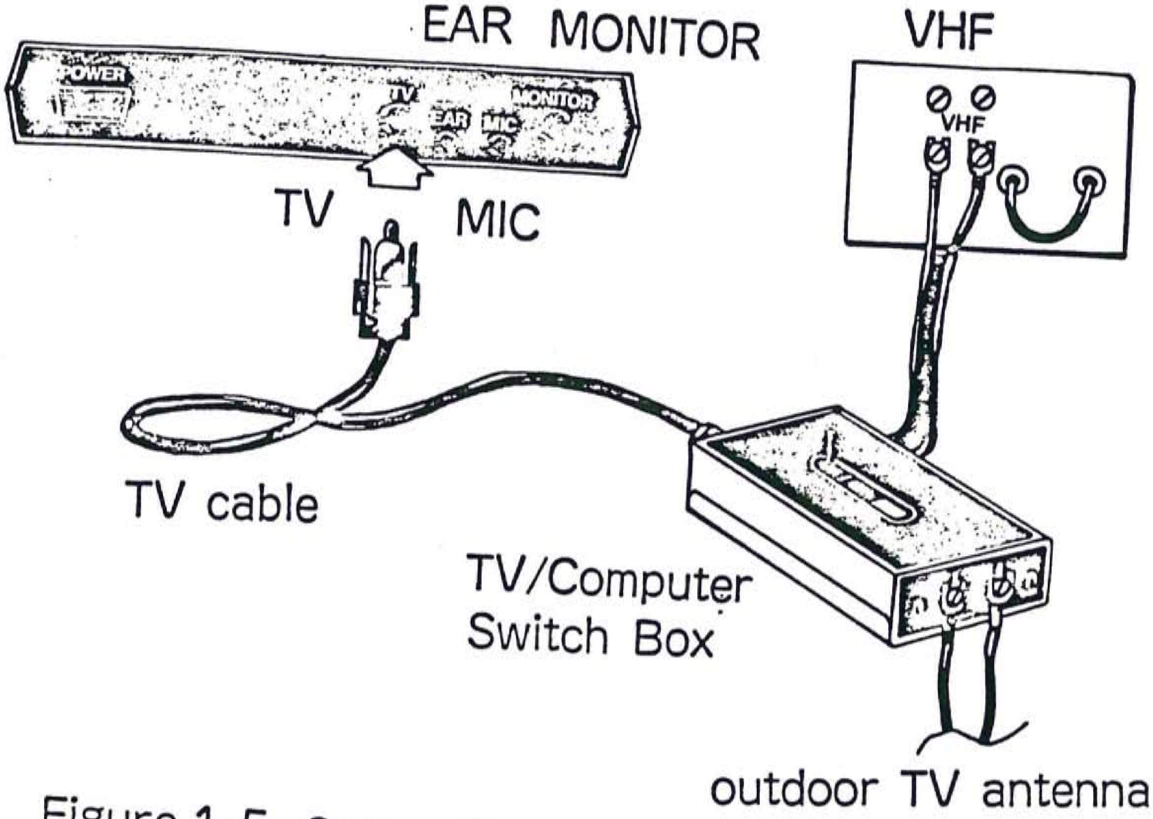


Figure.1-5 Connecting TV

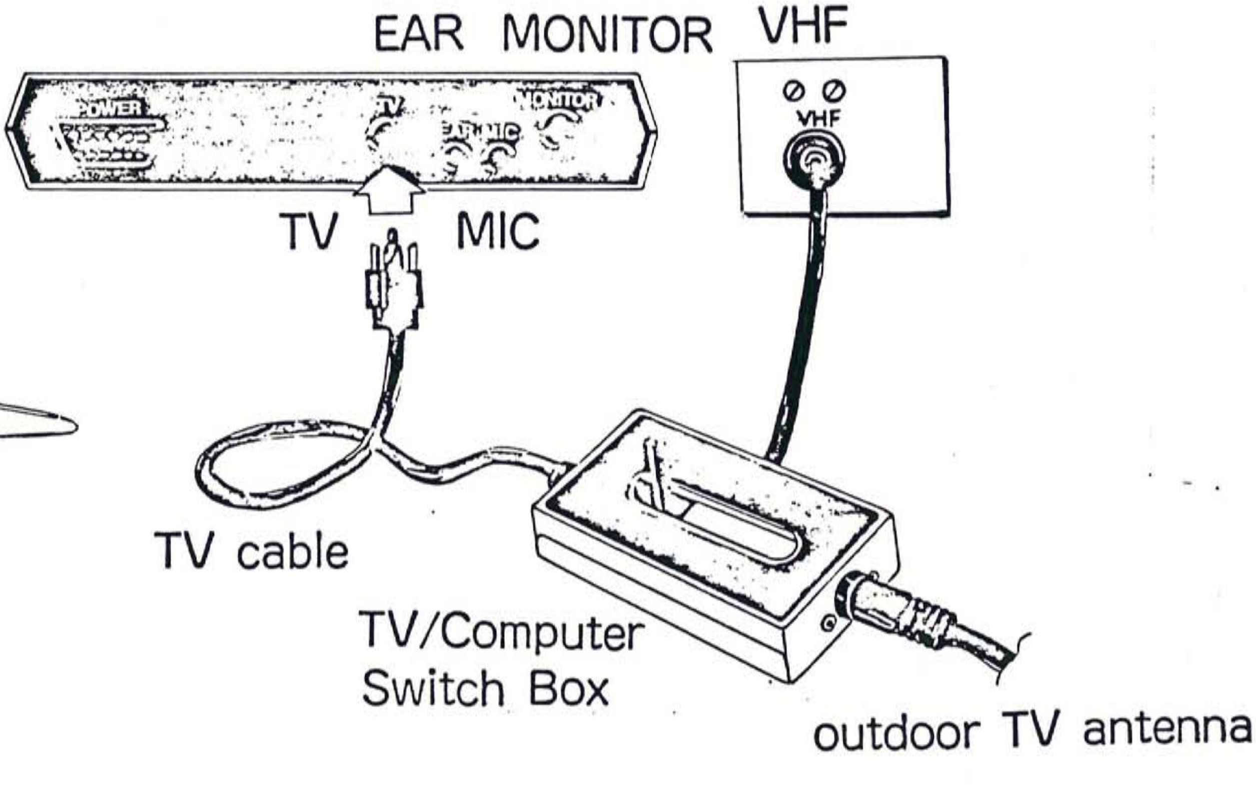


Figure.1-5 Connecting TV

1 • 3 The Power Supply

1. Plug the power cord of the Switching Power Supply into the wall outlet. (Fig.1-6)
2. Turn the Switching Power Supply on, the red power-on LED indication light will illuminate.
3. Turn the Switching Power Supply off.
4. Connect the DC output line of the Switching Power Supply to your MPF-II. (Fig.1-6)
5. Turn on the Switching Power Supply, your MPF-II will beep.
6. If it does not beep, press the RESET key; if it beeps after the RESET key was released, the power is supplied properly.
7. If the MPF-II does not beep, repeat the procedure from the beginning.

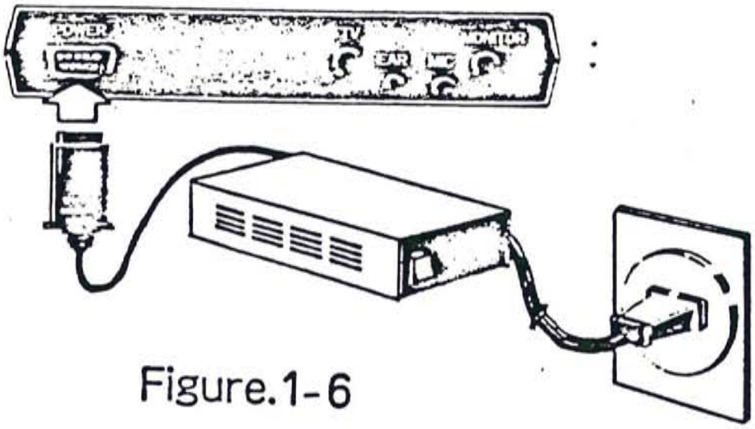


Figure.1-6

1 • 4 Test

1. Set the television on the channel 13; if your TV set only has push-button channel selector, please adjust the selector until the screen clearly and stably shows:

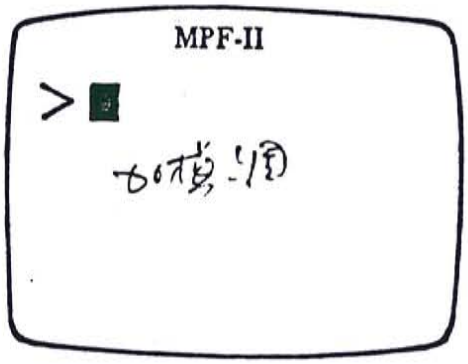


Fig. 1-7

不要忘了把中文去掉

> and █ are tow special signs the computer uses to communicate with us. ">" indicates the beginning of a line, and is called a prompt or a prompt character. "█" reminds a user that the computer is ready to receive a character. It is referred to as a cursor.

2. Now, key in the following program:

```
10 GR
20 FOR I=0 TO 7
30 COLOR = I
40 VLIN 0,47 AT 2*I
50 NEXT I
RUN
```

(This is a BASIC program, if you have any question, consult the Microprofessor-II dealer in your area.)

3. After you have executed the above program, your TV screen/video display should show seven vertical lines in green, purple, white, green, orange, blue and white.

4. Now, the MPF-II has been installed correctly. Let's begin to test the memory space.

5. Enter the command NEW, and then press the RETURN key; key in ?FRE(Ø), and then press the RETURN key, your MPF-II will show the number "-26629". (The unmber -26629 is only approximate number.) If your MPF-II displays a number close to the above number, then the memory of your MPF-II is all right.

1 • 5 Attaching the Software Cartridge

1. Always turn off the switching power supply to prolong the life of your MPF-II and protect the electronic components when inserting or removing the software cartridge.
2. Plug the software cartridge into your MPF-II properly. (Fig. 1-9)
3. Turn on the switching power supply, your MPF-II will execute the program in the software cartridge immediately.

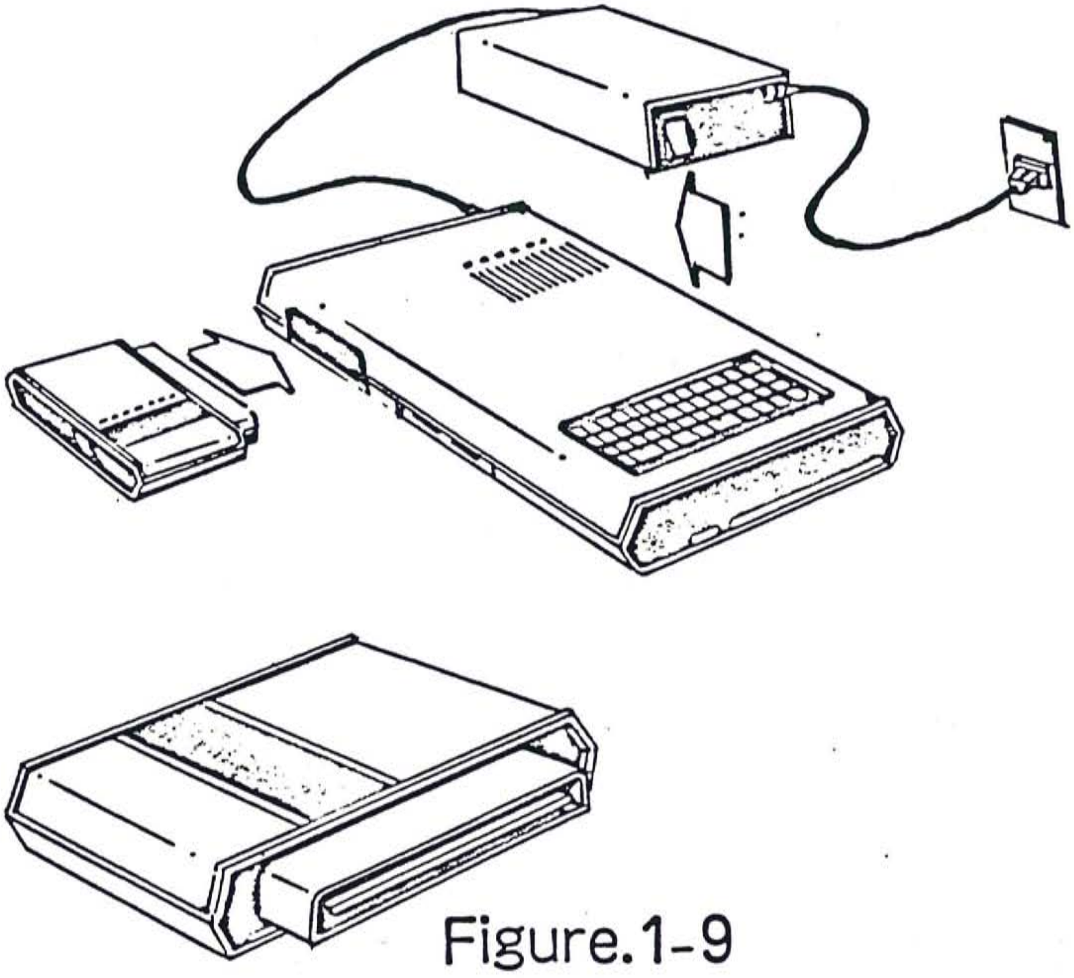


Figure.1-9

Installing the Software Cartridge

1 • 6 Connecting the Cassette Recorder

1.6.1 Load data from the cassette recorder

a. Get a recorder line, connect one end of it to the miniature jack marked "EAR" or "EXT SP" on the recorder and the other end to the jack marked "EAR" on the MPF-II. Then set the volume control of your recorder over middle loudness. See Fig. 1-1Ø

Note if you use either of the following two types of tape recorder, you can use the attached MPF-II recorder line.

* Single channel (MONO) recorder

* Two channel stereo recorder with two discrete phone jacks marked "EAR" or "EXT SP". You can plug the recorder line to any of the jacks marked "EAR" or "EXT SP".

* If your stereo recorder has only one jack for sound output, you can not use the attached recorder line. You have to use the recorder line especially designed for use with stereo recorders.

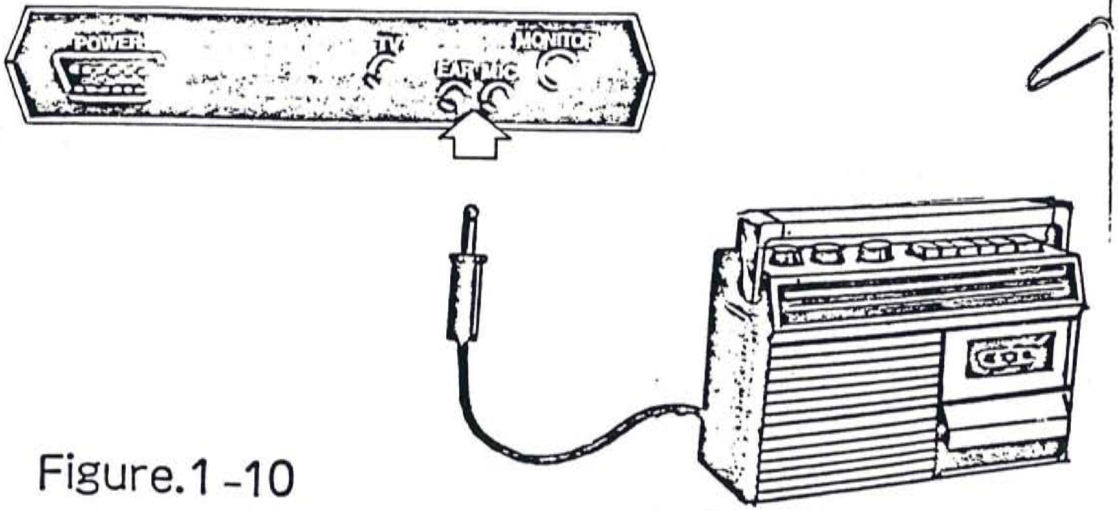


Figure.1 -10

Connectting the Cassette Recorder

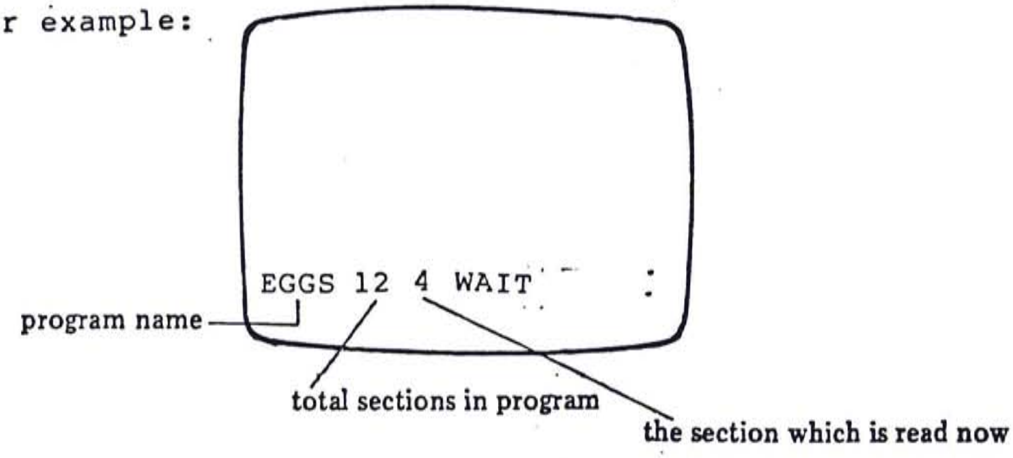
b. Put the tape into the recorder, rewind it to the starting point.

c. Key in "LOADT" if there is a filename for the data, type it in the quotation mark " ". If the program stored on tape doesn't have a filename, proceed to the next step.

d. Play the recorder, then press the RETURN key.

e. As soon as the MPF-II begins reading the tape, the screen will display the state of the tape loading operation.

For example:



```
EGGS 12 4 WAIT
```

f. If there is more than one file on a tape, you can get the file needed by information showed on the screen. Press FF(fast forward) key on the recorder, then the PLAY key repeatedly to find the desired file.

g. If the file is not loaded successfully, check: 1) if the recorder line is properly connected, 2) if the volume of the recorder is properly set, and then start the procedure over.

h. If the tape (of MPF-II fomated tape) is loaded successfully, the screen will display an "OK". Now you may press RUN and RETURN to execute the program already loaded into the memory of the MPF-II.

i. When any error happened, check if the connection is made properly.

j. As soon as it finished reading the data, the screen will display OK. Now, key in RUN, the MPF-II will begin to execute the program.

1.6.2 Store the data in the cassette

a. Connect the recorder line from the hole "MIC" on the recorder to the hole "MIC" on the MPF-II.

b. Press RECORD, PLAY on the recorder.

   Note if you use either of the following two types of tape recorder, you can use the attached MPF-II recorder line.

   * Single channel (MONO) recorder

   * Two channel stereo recorder with two discrete phone jacks marked "EAR" or "EXT SP". You can plug the recorder line to any of the jacks marked "EAR" or "EXT SP".

   * If your stereo recorder has only one jack for sound output, you can not use the attached recorder line. You have to use the recorder line especially designed for use with stereo recorders.

c. If the data is written in BASIC, key in SAVET, "   " (you may type in the filename in the quotation mark.), and then RETURN.

d. If the data is not written in BASIC, key in CALL-159 then type in as follows:

   starting address . ending address W filename, RETURN

e. Once the data was stored successfully onto the tape, the MPF-II will beep and show "OK" on the screen.

1.7 Joystick (JSK)

1. The optional joystick is used for game applications. You may connect the joystick to the MPF-II in accordance with the illustration below.

Fig. 1-1

2. The joystick is used to control the directions of game movements. Four directions are allowed. The button is equivalent to the ⊡ key on the MPF-II keyboard, which is used for shooting in most game applications.

3. The schematic of the joystick is provided in Appendix.

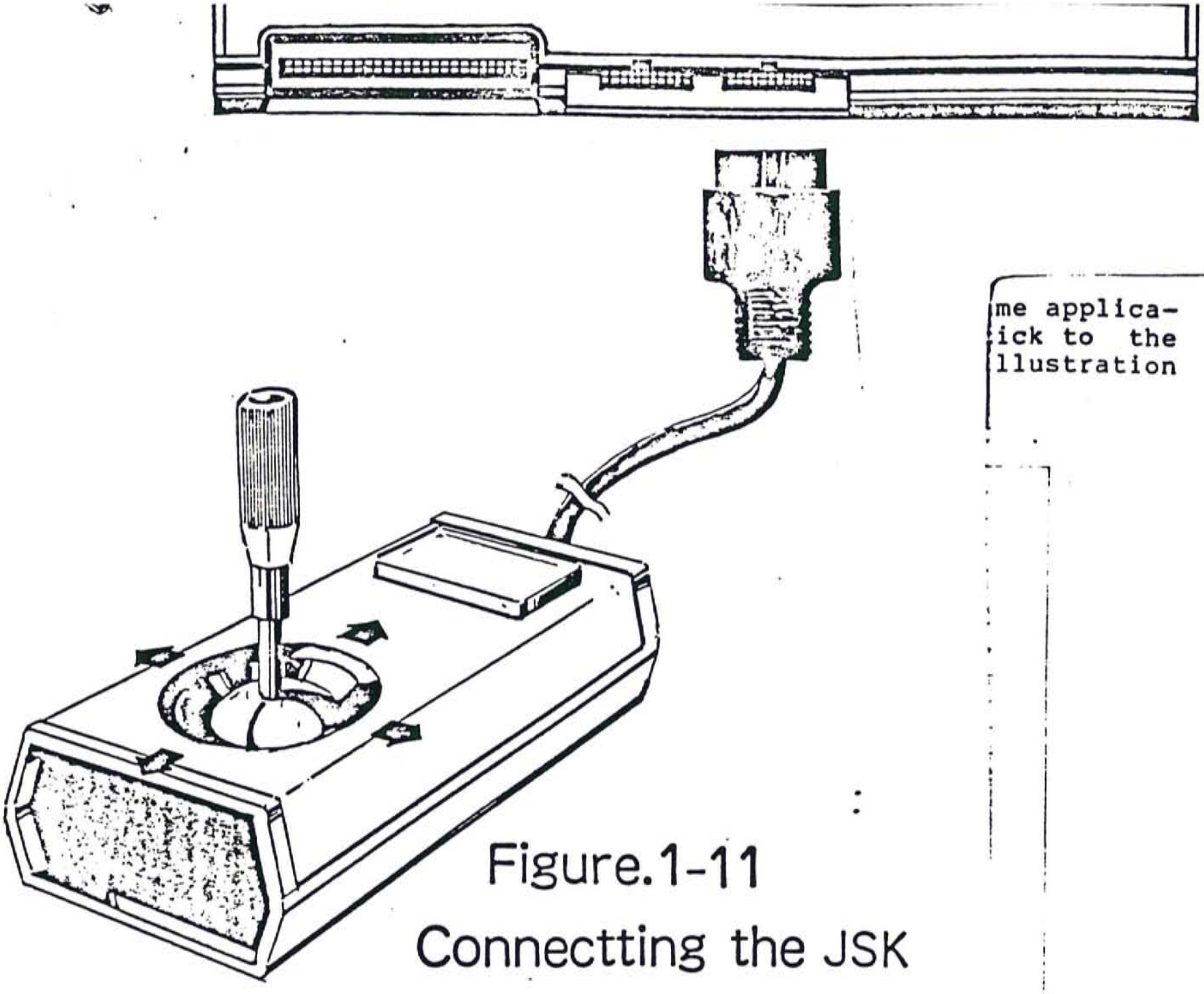


Figure.1-11
Connectting the JSK

me applica-
ick to the
llustration

directions of game movements. Four directions are allowed. The button is equivalent to the ⊡ key on the MPF-II keyboard, which is used for shooting in most game applications.

3. The schematic of the joystick is provided in Appendix.

1.8 Full-Size Keyboard

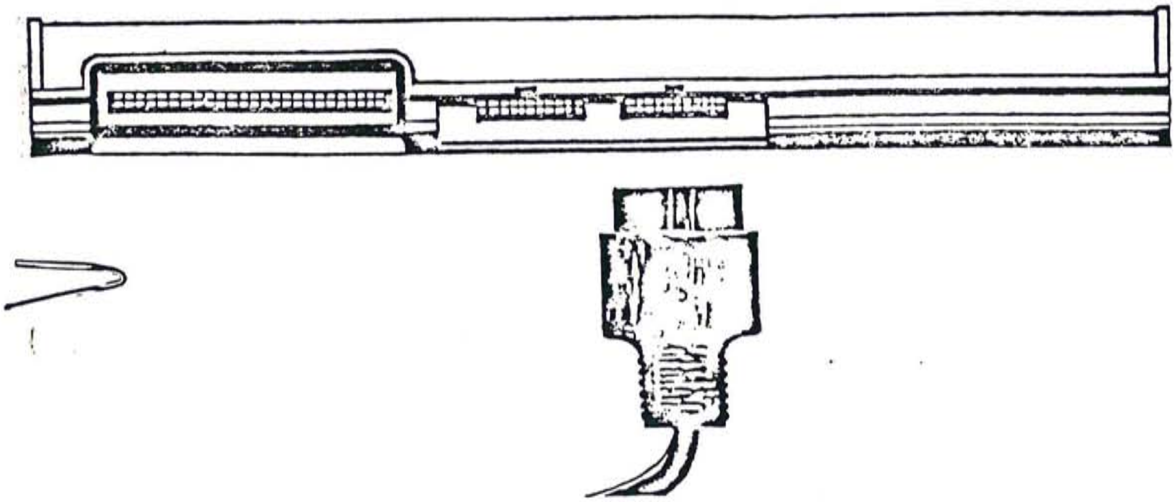
1. The optional full-size keyboard is designed for those who prefer standard typewriter keyboard. You may connect the full-size keyboard to the MPF-II in accordance with the illustration below.

Fig 1-12

Fig. 1-12 Full-size Keyboard

2. The specifications of the MPF-II are the same as the standard MPF-II keyboard except that the full-size keyboard has 55 keys, while the standard MPF-II keyboard has only 49 keys. This is because six more keys are added on the full-size keyboard for the users to use the keyboard more conveniently. The six keys and their corresponding keys on the standard MPF-II keyboard are listed in the table below.

On FSK-MPF	On MPF-II
FIRE x 2	⊡ x 1
SHIFT x 2	SHIFT x 1
← x 2	← x 1
← x 2	→ x 1
UP x 1	Nil



Note that the uP key is only for decoration. Still, there are four keys marked differently on the full-size keyboard. The four keys and their corresponding keys on the standard MPF-II keyboard are listed in the following table.

On FSK-MPF	On MPF-II
RETURN	↵
SPACE	▭
CTRL	CONTROL
FIRE	•

3. The schematic of the full-size keyboard is provided in Appendix.

1·9 The Printer

1. The printer with connector line for MPF-II is optional. Connect it to your MPF-II as the following figure shows.

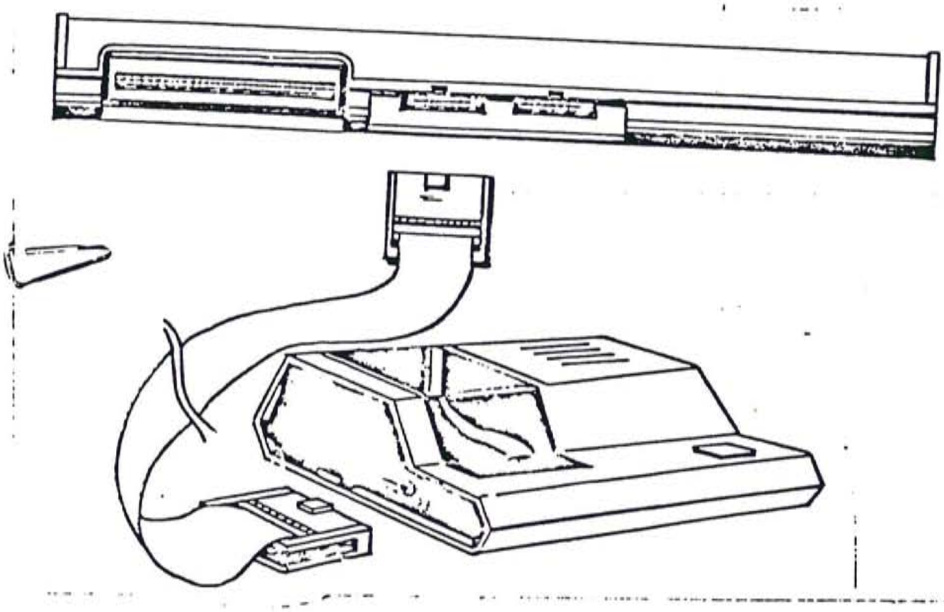
Fig.1-3

Fig.1-13

2. Turn on the power of the printer.

3. To copy the characters or graphics from screen, you can hold down the CONTROL key while pressing the P key or type HC directly on the screen.
   Note: HC stands for Hard Copy. This command enables you to print what's on the screen on paper.

4. Three commands for the printer in BASIC:
   a. PRTON - Print out everything after each PRINT command both on the TV screen and the printer.
   b. PRTOFF - After typing in this command, each time you use the PRINT command, information will be printed on the screen.
   c. HC - Copy the characters or graphics from TV screen to the printer.



CHAPTER 2

HARDWARE DESCRIPTION

2 • 1 Introduction to MPF-II Hardware

The major components of the MPF-II are switching power supply, the MPF-II main board, keyboard, and video display.

2.1.1. The specifications of the MPF-II are listed as follows:

1) The central processing unit (CPU):
   The CPU of the MPF-II is an R65Ø2 microprocessor.

2) The read only memory (ROM):
   The ROM capacity of the MPF-II totals 16K byte. It contains system monitor program and BASIC interpreter.

3) The read/write memory (random access memory):
   It is used to store user's program or data. (MPF-II contains 64K dynamic RAM).

4) Video display:
   The video display (either a monitor or a TV) has three modes of operation--text mode, low resolution graphics mode, and high resolution graphics mode.

5) Keyboard:
   The MPF-II's keyboard has 49 keys, including alphanumerical keys (A to Z, Ø to 9), function keys, and special signs such as #, $,...

6) EAR and MIC jacks:
   The ear and microphone jacks are used to store or retrieve data into (from) cassette tapes.

7) Edge connector for ROM (software) cartridges:
   A socket at the upper left side of the MPF-II is reserved for software cartridges. When a ROM cartridge is connected to the MPF-II, the MPF-II will execute the program stored in the software cartridge.

8) Printer edge connector:
   A printer socket is reserved for interfacing to printers with Centronics interface for making hard copies.

9) Edge connector for joystick and full-size keyboard:
A socket is reserved for either a joystick or a full-size keyboard.

1Ø) An 8 ohm, 1/4W speaker is built inside the MPF-II cabinet.

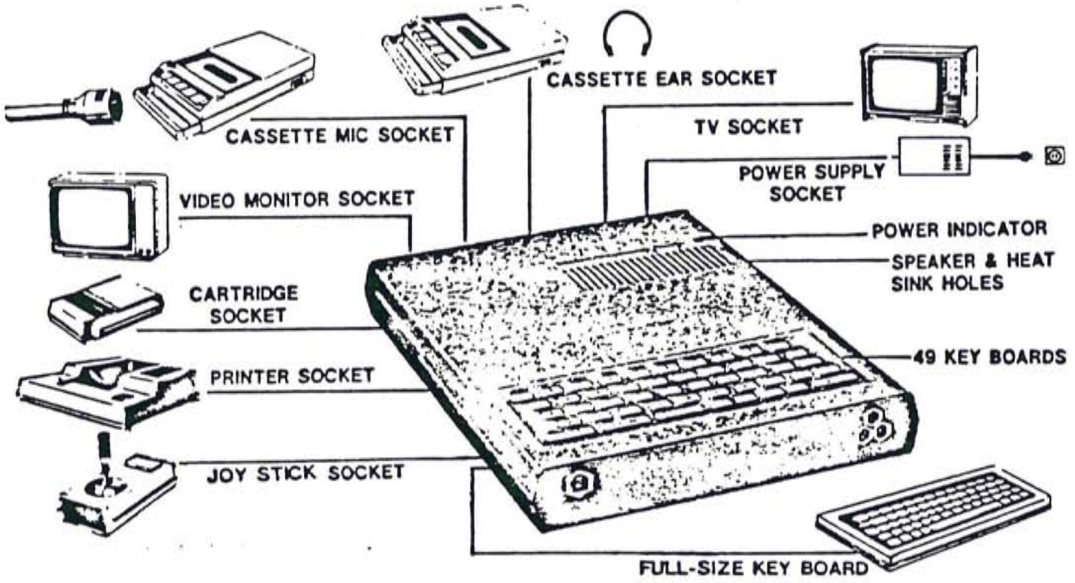
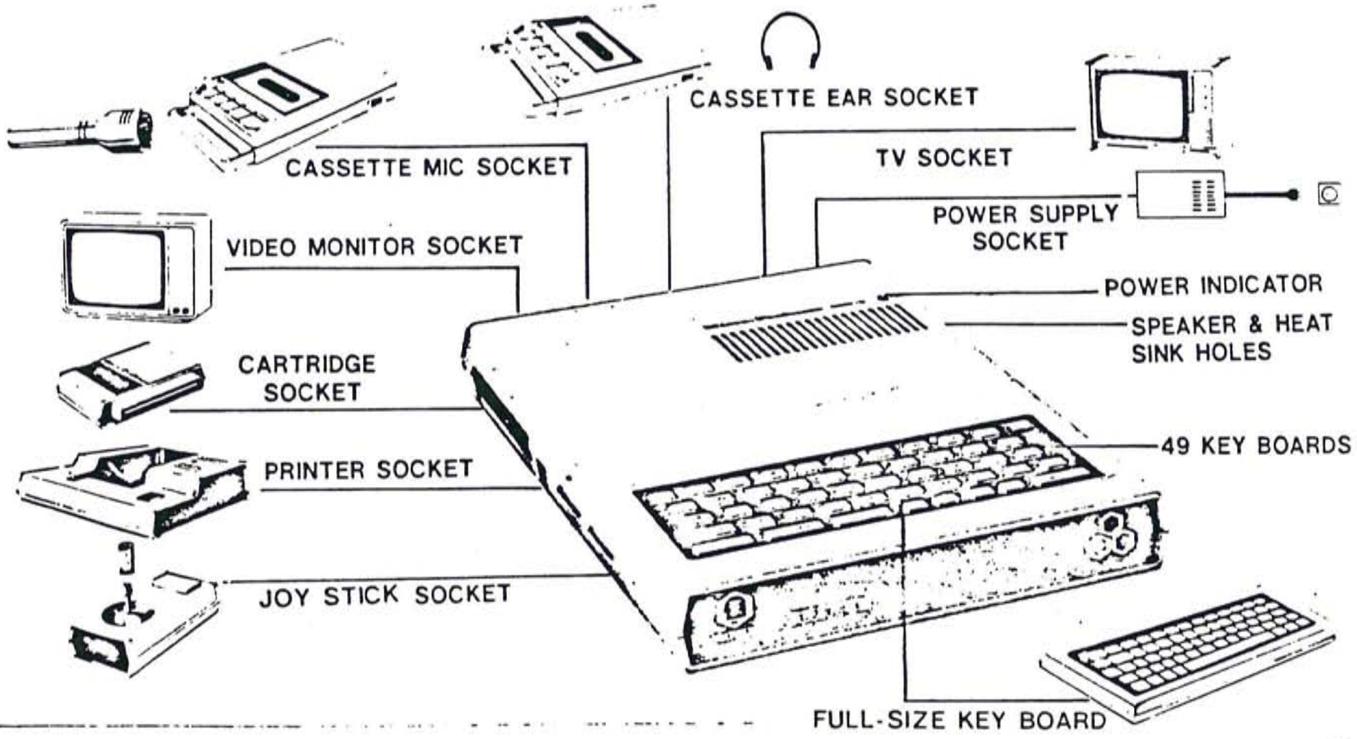
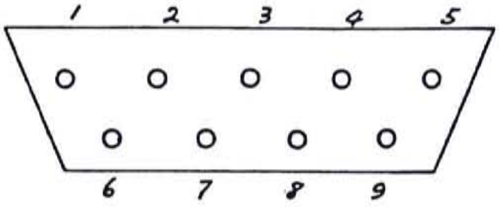


Fig. 2-1 The Micro-Professor II



2•2 The Power Supply

Three voltages are supplied to the MPF-II: +5V, -5V, and +12V. The power supply of the MPF-II is of switching type. The input voltage is 11ØVAC or 22ØVAC, and the output power are: +5V, 3.Ø A; -5V, 1ØØ mA; and +12V, 25Ø mA. Fig.2-2 shows the pin assignments of the power supply connector.



Pin 1 supplies +12V
Pin 2,3,4 supply +5V
Pin 5 supplies -5V
Pin 6,7,8,9 are for ground

Fig. 2-2 The Pin functions of the Power Supply

2 • 3 The Main Board

Opening the MPF-II cabinet, you will see a dark green printed circuit board (PCB). It is the MPF-II microcomputer itself. About 5Ø integrated cir-cuits (IC) are installed on the main board. Among the ICs, there is a 4Ø-pin R65Ø2, which is the brain of the MPF-II. It is capable of performing more than 5ØØ,ØØØ additions or subtractions per second. It is one of today's most widely used microprocessors.

The 65Ø2 is an 8-bit microprocessor, which means it handles data eight bits at a time. It is capable of accessing 64K bytes (65,536 bits) of data and operates on an instruction set containing 56 instructions. This microprocessor has 13 addressing modes through which it accesses the data stored in memory.

To the right of the ROM cartridge edge connector, two 28-pin ICs are installed side by side. The two ICs are for storing:

1) monitor programs
2) BASIC interpreter

Below the leftside of the two 28-pin ICs, there are eight 16-pin ICs. These ICs are used as the RAM of the MPF-II, which can store 64K of bytes of data (depending on the model you own). You are actually write your programs or data unto the RAM of your MPF-II when you write a program (or data) or run your program. Note that when the power of the MPF-II is turned off, all the data stored in the RAM is lost.

Through the ICs and other components of the MPF-II, you and the MPF-II can communicate with each other. After the MPF-II has processed the programs or data, it will display the results on the video monitor. By looking at what's displayed on the video monitor or TV, you will know what the MPF-II is doing. At the upper leftside of the MPF-II, there is a 5Ø-pin peripheral connector (also called software cartridge edge connector, which can be used to connect with peripheral devices other than ROM cartridges. For example, a floppy disk drive can be connected to the MPF-II through the peripheral connector.

2 • 4 Keyboard

A user has to key in his program or data using the keyboard of the MPF-II.

2.4.1. Keyboard specifications:

The MPF-II keyboard consists of 49 keys, which can generate 153 ASCII codes. The 153 ASCII codes (in hexadecimal) are listed in table 2-1.

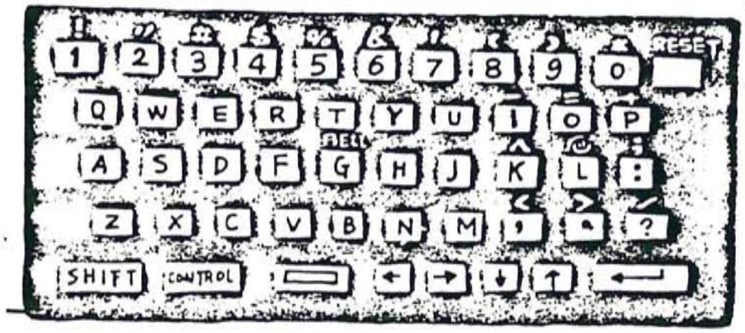
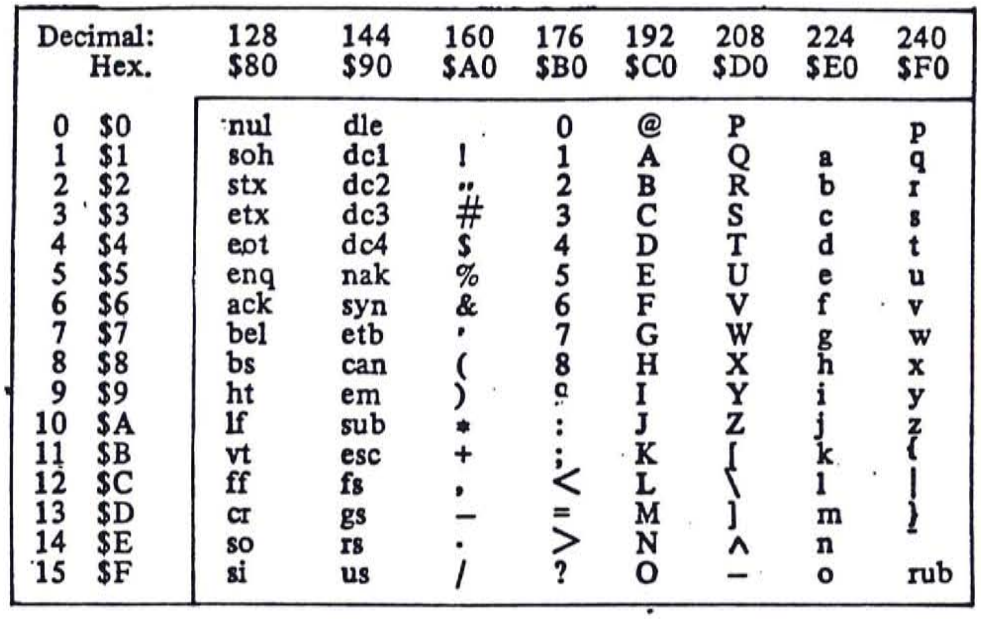


Fig. 2-3 MPF-II Keyboard

Key	Alone	CTRL	SHIFT	CTL-SHT	Key	Alone	CTRL	SHIFT	CTL-SHT
Space	A0	A0	A0	34	J	CA	8A	F2 5F	F2 5C
0 •	B0	B0	AA	35	K ^	CB	8B	DE	41
1 !	B1	B1	A1	F2 41	L @	CC	8C	C0	42
2 -	B2	B2	A2	F2 42	M	CD	8D	F2 72	F2 5D
3 #	B3	B3	A3	F2 43	N	CE	8E	F2 63	F2 53
4 $	B4	B4	A4	F2 44	O -	CF	8F	BD	43
5 %	B5	B5	A5	F2 45	P +.	D0	90	AB	44
6 &	B6	B6	A6	F2 46	Q	D1	91	F2 64	F2 48
7 ,	B7	45	A7	F2 47	R	D2	92	F2 67	F2 4B
8 (	B8	B8	A8	36	S	D3	93	F2 6D	F2 55
9)	B9	B9	A9	37	T	D4	94	F2 5E	F2 4C
: ;	BA	BA	BB	38	U	D5	95	F2 70	F2 4E
, <	AC	AC	BC	39	V	D6	96	F2 6B	F2 5B
. >	AE	AE	BE	3A	W	D7	97	F2 65	F2 49
? /	BF	BF	AF	3B	X	D8	98	F2 6F	F2 57
A	C1	B1	F2 6C	F2 54	Y	D9	99	F2 71	F2 4D
B	C2	82	F2 62	F2 52	Z	DA	9A	F2 6E	F2 56
C	C3	83	F2 6A	F2 5A	RETURN	8D	8D	8D	40
D	C4	84	F2 63	F2 58	←	88	88	88	3C
E	C5	85	F2 66	F2 4A	→	95	95	95	3D
F	C6	86	F2 69	F2 59	↓	F1	F1	F1	3E
G	C7	87	F2 60	F2 50	↑	F0	F0	F0	3F
H	C8	88	F2 61	F2 51	SHIFT	FF	FF	FF	FF
I –	C9	89	AD	F2 4F	CONTROL	FF	FF	FF	FF

Table 2-1 ASCII Code and their Corresponding Keys

The CONTROL key is used to generate control codes to the MPF-II, while the ←, →, ↓, ↑ keys are used for moving cursor. To find the decimal ASCII codes, use table 2-2.



Decimal:		128	144	160	176	192	208	224	240
	Hex.	$80	$90	$A0	$B0	$C0	$D0	$E0	$F0
0	$0	nul	dle		0	@	P		p
1	$1	soh	dc1	!	1	A	Q	a	q
2	$2	stx	dc2	"	2	B	R	b	r
3	$3	etx	dc3	#	3	C	S	c	s
4	$4	eot	dc4	$	4	D	T	d	t
5	$5	enq	nak	%	5	E	U	e	u
6	$6	ack	syn	&	6	F	V	f	v
7	$7	bel	etb	'	7	G	W	g	w
8	$8	bs	can	(	8	H	X	h	x
9	$9	ht	em	)	9	I	Y	i	y
10	$A	lf	sub	*	:	J	Z	j	z
11	$B	vt	esc	+	;	K	[	k	{
12	$C	ff	fs	,	<	L	\	l	|
13	$D	cr	gs	—	=	M	]	m	}
14	$E	so	rs	.	>	N	^	n	
15	$F	si	us	/	?	O	—	o	rub

Table 2-2 ASCII Character Set

In table 2-2, the abbreviations of ASCII control characters are represented in two and three lower case letters. But not all characters present in the ASCII character set can be generated by the MPF-II keyboard. For example, the two columns of lower case characters on the left, _, [, and control characters such as fs, us, rub can not be used on the MPF-II.

To obtain the decimal or hexadecimal value of any characters listed in the above table, you can add the decimal or hexadecimal numbers on the leftmost column to the numbers over the column in which a character appears.

The functions of the keys

1) RESET: The RESET key itself does not generate key code. It is directly connected to the microprocessor 6502. When this key is pressed, the MPF-II stops processing. As soon as this key is released, the MPF-II is initialized and starts a reset cycle.

   The SHIFT and CONTROL keys: The two keys do not generate key codes either. But when using together with other keys, the two keys will generate useful key codes.

2) Refer to the schematics marked with SHEET 1 of 1, MPF-II KEYBOARD. If only the alphabetical keys was pressed, the codes generated by the keyboard are shown on the upper right part of each circle.

3) The pressing of alphanumeric keys while holding down the SHIFT key will generate key codes shown on the upper left part of the circles.

4) The pressing of alphanumeric keys while holding down the CONTROL key will generate the key codes on the lower left part of the circles.

5) Holding down the SHIFT, CONTROL, and alphanumeric keys will generate the key codes shown in the lower right part of the circles.

6) The MPF-II uses 50 special signs (please refer to the graphics nameplate packaged together with the MPF-II). These special signs can be used in your programs for displaying special shapes. But these special signs can not be used for arithmetic operations. Before keying in these special signs, you have to press CONTROL and B simultaneously, then you can type in the special signs in accordance with the principles set forth in rules 3) and 5). When you have finished keying in the special signs, be sure to type CONTROL and B simultaneously to return to normal keyboard function.

2.4.3 Single Keystroke BASIC Commands

Normally each BASIC command consists of several English characters such as PRINT, IF...THEN, GOTO etc. Each time you type in a command, you have to press several keys. For your convenience, the single keystroke BASIC command feature was incorporated into the MPF-II. With this feature, you only have to press a key to enter a BASIC command. (Please refer to the diagram below and the attached SINGLE KEYSTROKE BASIC COMMAND NAMEPLATE.)

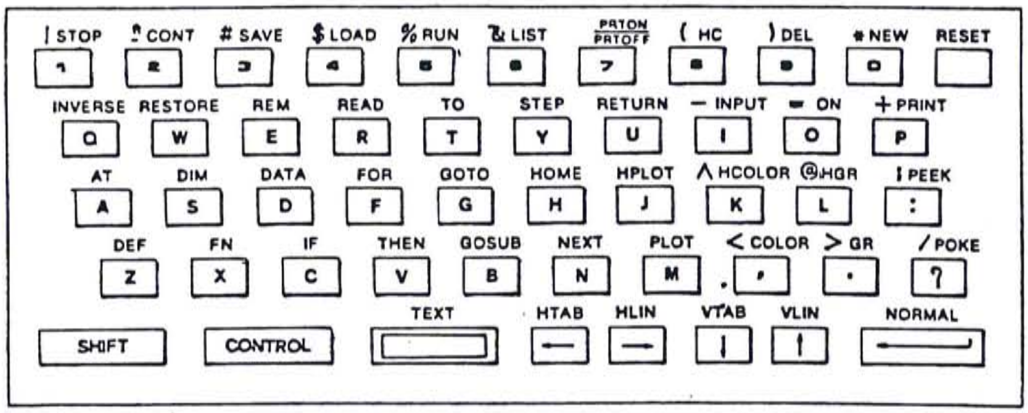


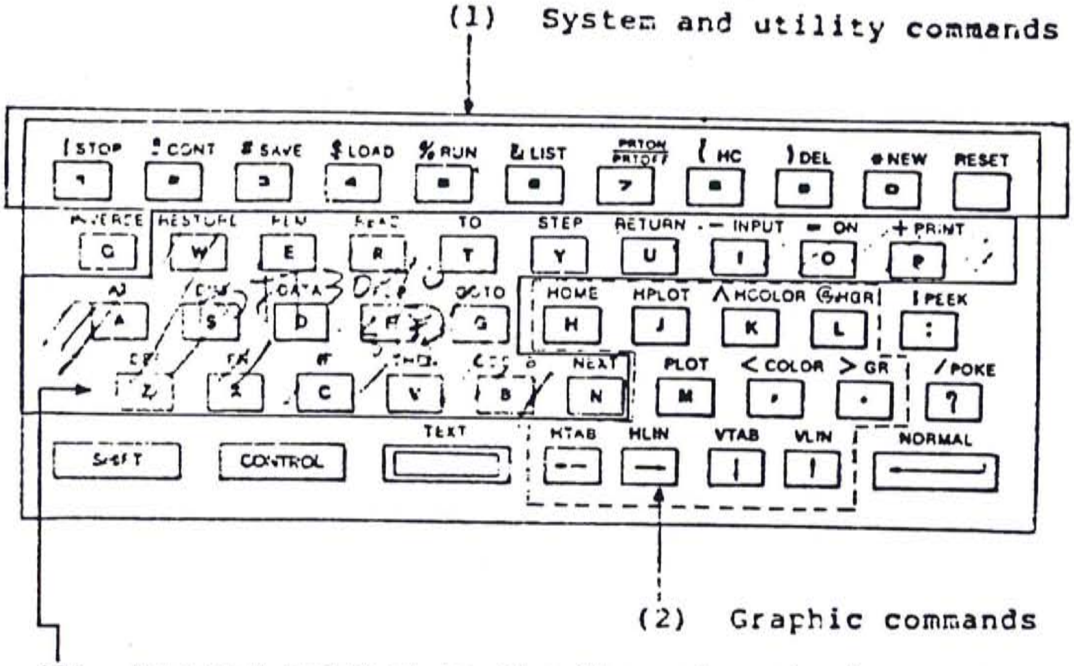
Fig 2-4 Nameplate for single keystroke cammands

To enter a single kdystroke BASIC command, you have to press the single keystroke BASIC command keys while holding down the SHIFT and CONTROL keys. Note that the $\frac{\text{PRTON}}{\text{PRTOFF}}$ is an on-and-off (toggle type) switch. Pressing this key once will change the status from PRINT-ON to PRINT-OFF or from PRINT-OFF to PRRINT-ON.

For your convenience, the keys used for entering single keystroke BASIC commands are so arranged as for you to access these keys easily. Related keys are grouped together. For example, the most frequently used system and utility commands are positioned in the upper row on the keyboard. By upper row, we mean the row on the keyboard that is farthest from you. Refer to Fig.2-5. The commands used for graphics can be entered by using the single keystroke BASIC command keys located at the lower right part of the keyboard. Keys used for commands related to the flow of control are located at the center of the keyboard, while Input/output commands are entered using the keys roughly bordering the keys related to the flow of control.

Keys used for entering single keystroke BASIC commands are usually paired. For example, the IF, THEN, GOTO, and GOSUB keys are paired close together so you can enter these commands conveniently. Other examples of the paired keys are LOAD, SAVE; READ, DATA; DEF,FN; STOP, CONTINUE, etc. It pays studying the nameplate for single keystroke BASIC commands just for a few minutes.

The function of the NORMAL command is just the opposite of that of the INVERSE. This is also reflected through the positioning of the keyboard. The NORMAL key is located at the lower right corner of the keyboard, while the INVERSE key at the upper left corner of the keyboard.



(1) System and utility commands

(2) Graphic commands

(3) Command related to the flow of control and Input/Output commands (enclosed in circles)

Fig 2-5 Nameplate for single keystroke cammands

2 • 5 Video Display

2.5.1 The Video Display Connector

When holding an MPF-II with the keyboard toward you, the side panel which is the farthest from you is referred to as the back panel on which you can see the connectors reserved for the power supply, ear and microphone jacks, and two holes marked TV and MONITOR, respectively.

A cable (sometimes known as RCA phone cable) is provided for connecting your MPF-II and the video display (which may be either a TV or a monitor). If you choose to connect your MPF-II with a TV, then you must find a switch box which is about two and a half inch long to connect to your video connector cable.

2.5.2 Display Modes

Three display modes are available for the MPF-II:

1) TEXT MODE: In text mode, the MPF-II has a display format of 24 rows by 40 columns of alphanumeric characters and special signs. Each character is comprised of a character font of 5 by 7 dot matrix. A one-dot wide space on either side of a character and above each character is used to keep the words apart.

2) LOW RESOLUTION GRAPHICS MODE: The MPF-II can display an array of 40 blocks wide and 48 blocks long. Each block may come in any of the six colors: black, white, blue, orange, purple, and yellow. No space is reserved between blocks. Thus, two adjacent blocks form a bigger block.

3) HIGH RESOLUTION GRAPHICS MODE: The MPF-II provides an array of 280 dots wide and 192 dots high. Each dot has the same size as that in text mode. Each dot may come in any of the six colors provided also in low resolution graphics mode.

2.5.3 Video Screen Display Buffer

The source of information used in text, low resolution graphics, and high resolution graphics modes is stored in the same area.

2.5.4 Screen Pages

The source of information needed to form a screen display are actually stored in two areas whose size is exactly the same. One of them is called "primary page", and the other is called "secondary page". The secondary page is needed for storing screen information which you want to display instantly. But if your MPF-II is of the version with only 16K RAM, the primary page or "page 1" is actually the secondary page or "page 2". The memory locations used for screen display information are illustrated in Table 3.

Table 2-3 Screen Display Memory Locations

Page 1		Page 2	
Starting Address			
Hexadecimal	Decimal	Hexadecimal	Decimal
2000	8192	3FFF	16383
Ending Address			
A000	40960	BFFF	49151

2.5.5. Screen Switches

To decide which mode and page is to be used for screen display, we use screen switches. However, you can not touch and see these switches, because these switches are controlled by the software (program) of the MPF-II. Each switch corresponds to a specific memory location. Each time a program is used to turn on or turn off (toggle) a switch, it simply references the specific memory location of that switch. Though data is read from or written to that memory location each time the program references that memory location, it is the reference of that memory location that actually toggles the switch. Since these switches are controlled by the software, they are generally known as "soft switches".

There are four memory locations used as soft switches. They come in pairs. Therefore, when one switch is turned on, the other in the pair is always off. Table 4 lists the addresses of the switches:

Table 2-4 Screen Switches

Address			
Hex	Decimal		Functions
CØ5Ø	49232	-163Ø4	Display graphics mode
CØ51	49233	-163Ø3	Display text mode
CØ54	49236	-163ØØ	Display page 1
CØ55	49237	-16299	Display page 2

2 • 6 Screen Display Summary

2.6.1 Specifications of Video Display

1) Display Type: Memory mapped into the RAM
2) Display Mode: Text mode, low resolution graphics mode, high resolution graphics mode.
3) Text display format: 960 characters (24 rows by 40 columns)
4) Character font: 5 by 7 dot matrix
5) Character mode: Normal, inverse
6) Character set: 64 upper case ASCII characters
7) Graphics capability: 1920 blocks (40 by 48) for low resolution graphics, 53760 dots (280 by 192) for high graphics mode.
8) Number of colors: six colors for both low and high resolution graphics modes.

2.6.2 Text Mode

In text mode, the MPF-II is capable of displaying a screen of 24 rows and 40 columns. Characters displayed on the screen are stored in the video display buffer. Each character position corresponds to one memory location, which can be used to store any of the ASCII code of 64 characters--26 upper case English letters, 10 numerals, and 28 special characters. Each character is displayed as a 5 by 7 dot matrix with a one-dot space on both side of the character and above the character to keep the word apart.

Fig. 2-5 shows the 64 characters that can be displayed on the screen

```
◎ A B C D E F G H I J K L M N O
P Q R S T U V W X Y Z [ \ ]   —
  ! " # $ % & ' ( ) * + , - . /
0 1 2 3 4 5 6 7 8 9 : ; < = > ?
```

Fig 2-5 MPF-II Character Set

In low resolution graphics mode, the screen can display an array of 4Ø by 48 blocks, while each block may come in any of the six colors available on the MPF-II. But the blocks come in only white and grey on monochrome monitors.

Table 2-5 Low Resolution Graphics Colors

Decimal	Hex	Color	Decimal	Hex	Color
Ø	$Ø	black	8	$8	Purple
1	$1	green	9	$9	green
2	$2	purple	1Ø	$A	purple
3	$3	white	11	$B	white
4	$4	green	12	$C	white
5	$5	orange	13	$D	orange
6	$6	blue	14	$E	blue
7	$7	white	15	$F	white

If color does not show up on your color TV screen, you can adjust a screw (which connects to a variable capacitor on the MPF-II main board) at the bottom of the MPF-II until color shows up on your screen. The shade of color may vary, depending on different brands of color TVs. You may use the color trimmer on your screen to make adequate adjustment.

2.6.4 High Resolution Graphics

When your MPF-II operates in high resolution graphics mode, the screen displays an array of 53,76Ø dots (28Ø by 192). Each dot may come in one of the six colors--white, black, green, blue, orange, purple.

When operating in high resolution graphics mode, the MPF-II fetches screen information from a memory area consisting of 8,192 bytes. This memory area is known as display buffer. The display buffer is divided into two areas: page 1 and page 2. The memory range of page 1 or primary page is from 2ØØØH (hexadecimal) or 8192 (decimal) through 3FFFH or 16383, and the memory range of page 2 is from A000H (or 40960) through BFFFH (or 49151).

In text and low resolution graphics modes, the two memory areas of page 1 and page 2 are also used for screen buffer.

Each dot displayed on the screen (when the MPF-II operates in high resolution graphics mode) represents one bit in the screen buffer. The seven bits of a byte is displayed on the screen, while the remaining bit is used for selecting color for the seven bits displayed on the screen.

Each line on the screen requires 40 bytes of information. The first bit of the first byte is displayed as the leftmost bit (or at the leftmost position of a line), followed by the second bit, third bit through the seventh bit. The eighth bit of the first byte is used for selecting color, so it is not displayed. Following the seventh bit of the fisrt byte is the first bit of the second byte, and the first bit of the third byte follows the seventh bit of the second byte. Therefore, each line consists of

7 (dots) x 40 (bytes) = 280 (dots)

On a black-and-white TV or video monitor, if a dot whose corresponding bit is 1 or "on", the dot appears in white on the screen; if the dot whose corresponding bit is 0 or "off", the dot comes in black on the screen.

However, on a color TV or monitor, the color of a dot is determined not only by whether its corresponding bit is on or off, but also by the position where the dot appears on the screen. If the corresponding bit of the dot is off, the dot will come in black. But if the corresponding bit of the dot is on, the color of the dot will be decided by its position on a screen. If the dot appears on the column 0, leftmost column, or on even-numbered columns, it will come in purple. If it appears on column 279, rightmost column, or any of odd-numbered columns, it will appear in green. If two dots come side by side, they will be in white. If the dot which is contained in a byte with the eighth bit being one, then the purple and green colors will be replaced by blue and orange colors.

In high resolution graphics mode, there are six colors available. However, they are subject to the following limitations:

1) Dots on even-numbered columns only come in black, blue, or purple.

2) Dots on odd-numbered columns only come in black, green, or orange.

3) Each byte must come in either purple/green or blue/orange. It is absolutely impossible that purple/orange, purple/blue, green/orange, green/blue come in the same byte.

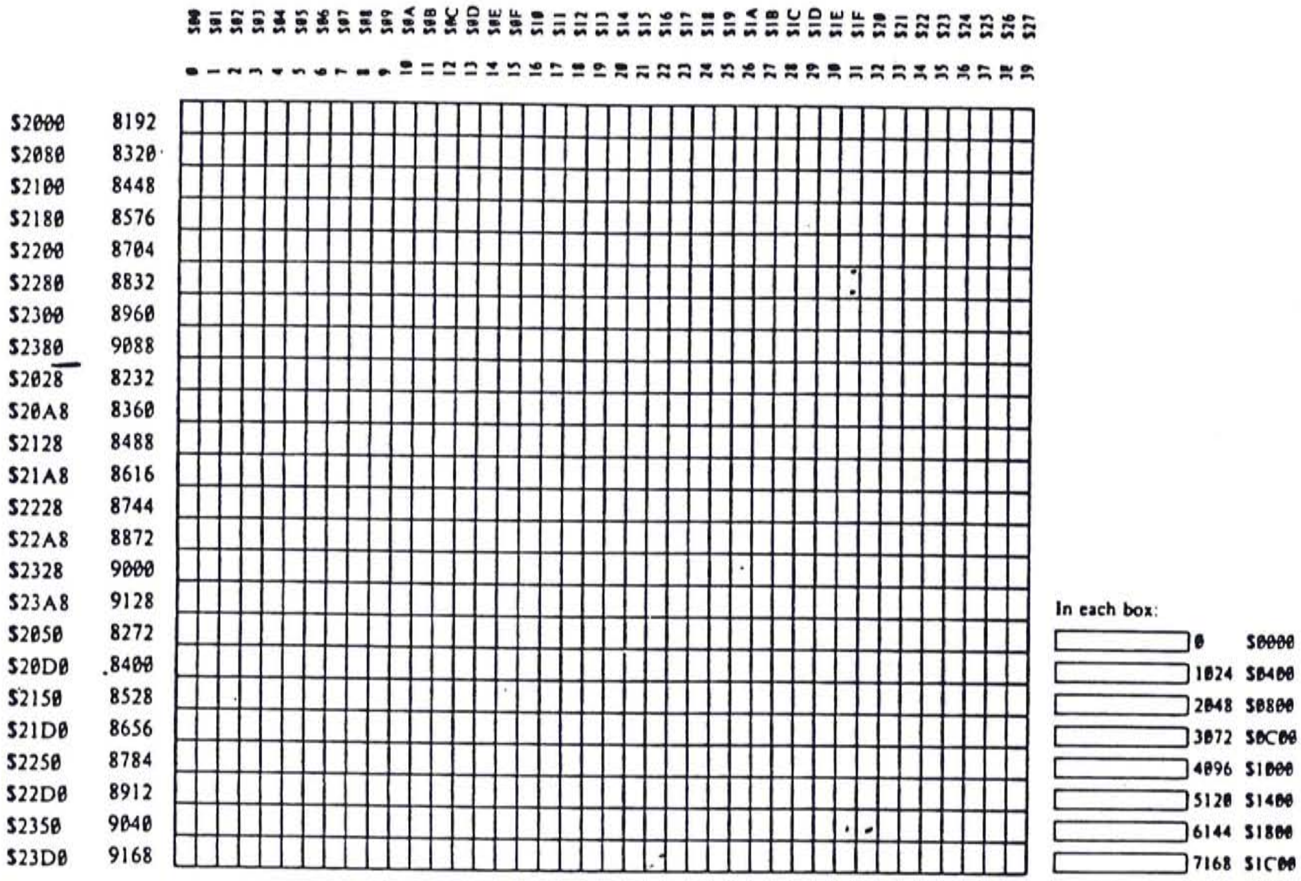
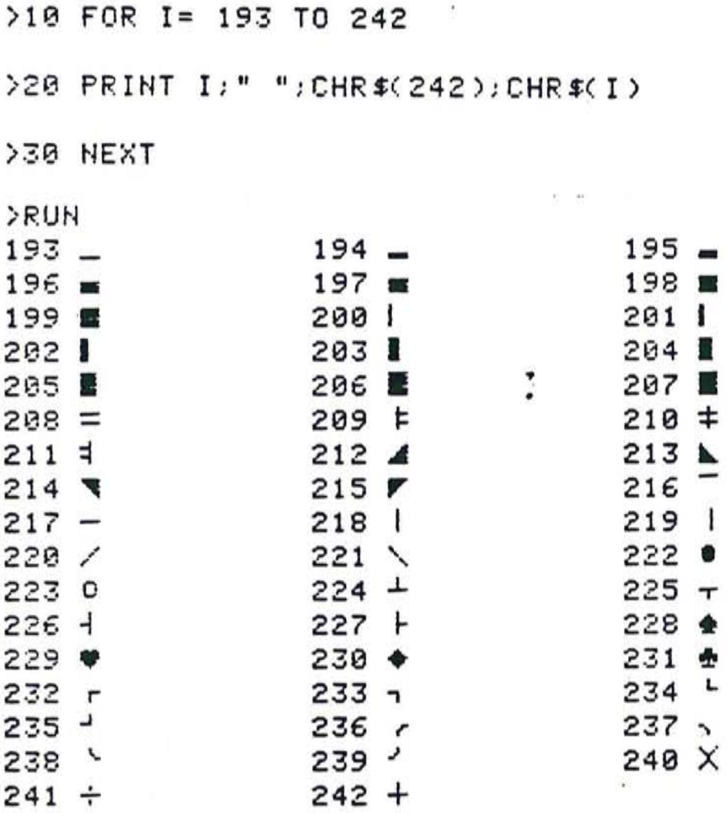


Fig. 2-6 Map of the High Resolution Graphics Mode

2.6.5 Graphic Pattern Display

The MPF-II can display 5Ø special patterns as shown on the "Special Pattern Keyboard Overlay". Type in the following program and execute the program, then the special patterns will be displayed on the screen. You can compose your own graphics and pictures, using these patterns.



```
>1Ø FOR I= 193 TO 242

>2Ø PRINT I;" ";CHR$(242);CHR$(I)

>3Ø NEXT

>RUN
193 _               194 ▂               195 ▃
196 ▄               197 ▅               198 ▆
199 ▇               2ØØ ▏               2Ø1 ▎
2Ø2 ▍               2Ø3 ▌               2Ø4 ▋
2Ø5 ▊               2Ø6 ▉               2Ø7 █
2Ø8 =               2Ø9 ╞               21Ø ǂ
211 ╡               212 ◢               213 ◣
214 ◥               215 ◤               216 ‾
217 —               218 |               219 |
22Ø /               221 \               222 ●
223 o               224 ┴               225 ┬
226 ┤               227 ├               228 ♠
229 ♥               23Ø ♦               231 ♣
232 ┌               233 ┐               234 └
235 ┘               236 ╭               237 ╮
238 ╰               239 ╯               24Ø X
241 ÷               242 +
```

Every pattern consists of eight bytes.

The special patterns displayed on the screen correspond to the value table for the special patterns (Table 2-6.) From the value table, it is obvious that each pattern is built up with eight bytes. Each group of 8-byte values is preceded by a hexadecimal number as an identifier. The hexadecimal identifiers start from Ø1H and end at 32H (H stands for hexadecimal). Therefore, there are 5Ø different groups of 8-byte values (32H = 5Ø decimal).

If we use the codes represented by the 2Øth group of 8-byte values (preceded by the hexadecimal identifier 14H) to generate a pattern, we have to find the eight values, using Table 2-6. The eight values are Ø, 4ØH, 6ØH,7ØH,78H,7CH,7EH,7FH. Of the eight bits of each byte, only seven bits--bit Ø through bit 6--will be displayed. Bit 7 of each byte in a group is used for selecting color. The pattern identified by 14H now looks like:

```
Bit Ø        Bit 7
  ↓            ↓
  Ø Ø Ø Ø Ø Ø Ø Ø<——The first byte
  Ø Ø Ø Ø Ø Ø 1 Ø<——The second byte
  Ø Ø Ø Ø Ø 1 1 Ø           .
  Ø Ø Ø Ø 1 1 1 Ø           .
  Ø Ø Ø 1 1 1 1 Ø           .
  Ø Ø 1 1 1 1 1 Ø           .
  Ø 1 1 1 1 1 1 Ø           .
  1 1 1 1 1 1 1 Ø<——The eighth byte
```

Note that bit 7 is the high order bit. Thus, the second byte is actually Ø1ØØØØØØ (binary), and the third byte is Ø11ØØØØØ (binary), etc.

Each bit represents a dot of a special pattern. When the content of a bit is "Ø", the bit will not be displayed on the screen. If the content of a bit is "1", it turns on a dot and the dot will be displayed. Now the illuminated dots will show on the screen as follows:

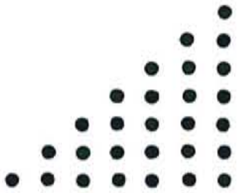


Table 2-6 shows the contents of the 50 bytes used for creating the shape table.

```
CRTABL:
;01
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    07FH
;02
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    07FH
          DEFB    07FH
;03
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    07FH
          DEFB    07FH
          DEFB    07FH
;04
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    07FH
          DEFB    07FH
          DEFB    07FH
          DEFB    07FH
;05
          DEFB    0
          DEFB    0
          DEFB    0
          DEFB    07FH
          DEFB    07FH
          DEFB    07FH
          DEFB    07FH
          DEFB    07FH
```

```
                DEFB    0
                DEFB    0
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
;07
                DEFB    0
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
                DEFB    07FH
;0B
                DEFB    01H
                DEFB    01H
                DEFB    01H
                DEFB    01H
                DEFB    01H
                DEFB    01H
                DEFB    01H
                DEFB    01H
;09
                DEFB    03H
                DEFB    003H
                DEFB    03H
                DEFB    003H
                DEFB    03H
                DEFB    003H
                DEFB    03H
                DEFB    003H
;0A
                DEFB    07H
                DEFB    07H
                DEFB    07H
                DEFB    07H
                DEFB    07H
                DEFB    07H
                DEFB    07H
                DEFB    07H
```

```
; 0B
            DEFB    0FH
            DEFB    0FH
            DEFB    0FH
            DEFB    0FH
            DEFB    0FH
            DEFB    0FH
            DEFB    0FH
            DEFB    0FH
; 0C
            DEFB    1FH
            DEFB    1FH
            DEFB    1FH
            DEFB    1FH
            DEFB    1FH
            DEFB    1FH
            DEFB    1FH
            DEFB    01FH
; 0D
            DEFB    3FH
            DEFB    03FH
            DEFB    3FH
            DEFB    03FH
            DEFB    3FH
            DEFB    03FH
            DEFB    3FH
            DEFB    03FH
; 0E
            DEFB    07FH
            DEFB    07FH
            DEFB    07FH
            DEFB    07FH
            DEFB    07FH
            DEFB    07FH
            DEFB    07FH
            DEFB    07FH
; 0F
            DEFB    000H
            DEFB    055H
            DEFB    02AH
            DEFB    07FH
            DEFB    0B0H
            DEFB    0D5H
            DEFB    0AAH
            DEFB    0FFH
```

```
;10
          DEFB    0
          DEFB    0
          DEFB    07FH
          DEFB    0
          DEFB    0
          DEFB    07FH
          DEFB    0
          DEFB    0
;11
          DEFB    08H
          DEFB    08H
          DEFB    78H
          DEFB    08H
          DEFB    08H
          DEFB    78H
          DEFB    08H
          DEFB    08H
;12
          DEFB    08H
          DEFB    08H
          DEFB    07FH
          DEFB    08H
          DEFB    08H
          DEFB    7FH
          DEFB    08H
          DEFB    08H
;13
          DEFB    08H
          DEFB    08H
          DEFB    0FH
          DEFB    08H
          DEFB    08H
          DEFB    0FH
          DEFB    08H
          DEFB    08H
;14
          DEFB    0
          DEFB    40H
          DEFB    60H
          DEFB    70H
          DEFB    78H
          DEFB    7CH
          DEFB    7EH
          DEFB    7FH
```

```
;15
            DEFB    0
            DEFB    01H
            DEFB    03H
            DEFB    07H
            DEFB    0FH
            DEFB    1FH
            DEFB    3FH
            DEFB    07FH
;16
            DEFB    0
            DEFB    07FH
            DEFB    7EH
            DEFB    7CH
            DEFB    78H
            DEFB    70H
            DEFB    60H
            DEFB    40H
;17
            DEFB    0
            DEFB    07FH
            DEFB    03FH
            DEFB    1FH
            DEFB    0FH
            DEFB    07H
            DEFB    03H
            DEFB    01H
;18
            DEFB    07FH
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
;19
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    07FH
            DEFB    0
            DEFB    0
            DEFB    0
```

```
;1A
            DEFB    0BH
            DEFB    0BH
            DEFB    0BH
            DEFB    0BH
            DEFB    0BH
            DEFB    0BH
            DEFB    0BH
            DEFB    0BH
;1B
            DEFB    10H
            DEFB    010H
            DEFB    10H
            DEFB    010H
            DEFB    10H
            DEFB    010H
            DEFB    10H
            DEFB    010H
;1C
            DEFB    0
            DEFB    40H
            DEFB    20H
            DEFB    10H
            DEFB    08H
            DEFB    04H
            DEFB    02H
            DEFB    01H
;1D
            DEFB    0
            DEFB    01H
            DEFB    02H
            DEFB    04H
            DEFB    08H
            DEFB    10H
            DEFB    20H
            DEFB    40H
;1E
            DEFB    0
            DEFB    1CH
            DEFB    3EH
            DEFB    3EH
            DEFB    3EH
            DEFB    3EH
            DEFB    1CH
            DEFB    0
```

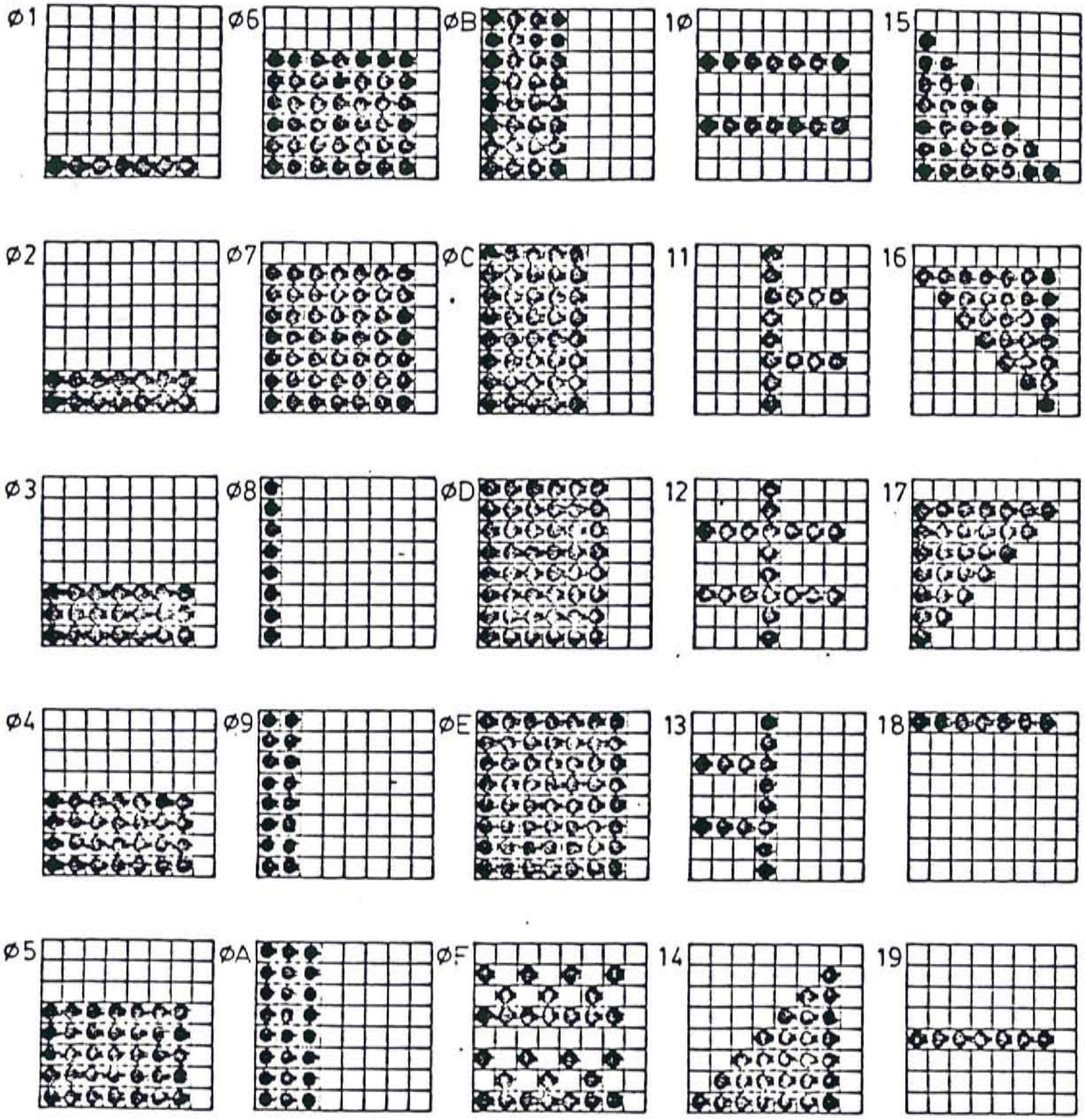
```
;1F
        DEFB    0
        DEFB    1CH
        DEFB    22H
        DEFB    22H
        DEFB    22H
        DEFB    22H
        DEFB    1CH
        DEFB    0H
;20
        DEFB    08H
        DEFB    08H
        DEFB    08H
        DEFB    08H
        DEFB    7FH
        DEFB    0
        DEFB    0
        DEFB    0
;21
        DEFB    0
        DEFB    0
        DEFB    0
        DEFB    0
        DEFB    07FH
        DEFB    08H
        DEFB    08H
        DEFB    08H
;22
        DEFB    08H
        DEFB    08H
        DEFB    08H
        DEFB    08H
        DEFB    0FH
        DEFB    08H
        DEFB    08H
        DEFB    08H
;23
        DEFB    08H
        DEFB    08H
        DEFB    08H
        DEFB    08H
        DEFB    78H
        DEFB    08H
        DEFB    08H
        DEFB    08H
;24
        DEFB    0
        DEFB    08H
        DEFB    1CH
        DEFB    3EH
```

```
                DEFB    7FH
                DEFB    1CH
                DEFB    3EH
;25
                DEFB    0
                DEFB    36H
                DEFB    7FH
                DEFB    7FH
                DEFB    7FH
                DEFB    3EH
                DEFB    1CH
                DEFB    08H
;26
                DEFB    0
                DEFB    08H
                DEFB    1CH
                DEFB    3EH
                DEFB    7FH
                DEFB    3EH
                DEFB    1CH
                DEFB    08H
;27
                DEFB    0
                DEFB    1CH
                DEFB    1CH
                DEFB    7FH
                DEFB    7FH
                DEFB    6BH
                DEFB    08H
                DEFB    3EH
;28
                DEFB    0
                DEFB    0
                DEFB    0
                DEFB    0
                DEFB    78H
                DEFB    08H
                DEFB    08H
                DEFB    08H
;29
                DEFB    0
                DEFB    0
                DEFB    0
                DEFB    0
                DEFB    0FH
                DEFB    08H
                DEFB    08H
                DEFB    08H
```

```
;2A
            DEFB    08H
            DEFB    08H
            DEFB    08H
            DEFB    78H
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
;2B
            DEFB    08H
            DEFB    08H
            DEFB    08H
            DEFB    0FH
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
;2C
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    60H
            DEFB    10H
            DEFB    08H
            DEFB    08H
;2D
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    3H
            DEFB    04H
            DEFB    08H
            DEFB    08H
;2E
            DEFB    08H
            DEFB    08H
            DEFB    10H
            DEFB    60H
            DEFB    0
            DEFB    0
            DEFB    0
            DEFB    0
;2F
            DEFB    08H
            DEFB    08H
            DEFB    04H
            DEFB    03H
```

```
                DEFB    0
                DEFB    0
                DEFB    0
                DEFB    0
;30
                DEFB    41H
                DEFB    22H
                DEFB    14H
                DEFB    08H
                DEFB    08H
                DEFB    14H
                DEFB    22H
                DEFB    41H
;31
                DEFB    0
                DEFB    08H
                DEFB    08H
                DEFB    0
                DEFB    07FH
                DEFB    0
                DEFB    08H
                DEFB    08H
;32
                DEFB    08H
                DEFB    08H
                DEFB    08H
                DEFB    08H
                DEFB    7FH
                DEFB    08H
                DEFB    08H
                DEFB    08H
```

Table 2-7 is the 5ø special patterns represented as illuminated dots.



2 • 7 Input/Output Expansion

2.7.1 Speaker

On the upper right part of MPF-II mainboard, there is a built-in speaker. You can program the speaker to generate various sounds.

The speaker is controlled by a "soft" switch. The switch can put the paper cone in two positions: "in" and "out". Every time a program references the memory location coresponding to the switch, the state of the speaker is changed. Each time the state of the speaker is changed, the speaker generates a tone. By changing the state of the speaker frequently and continuously, you can generate a tone from the speaker.

The memory location associated with the soft switch is 492ØØ or -16336 (decimal). The hexadecimal equivalent of that value is CØ3ØH. When this location is referenced, the speaker will generate a tone.

A program may reference this location by reading from or writing into the location. It is the "referencing" the location that change the state of the speaker. The value that is read from or written into the memory location has nothing to do with the flipping of the soft switch for speaker.

Note that when the 65Ø2 microprocessor performs a "write" operation, it must first perform a "read" operation. Therefore, if you reference the soft switch by writing a value into its associated location, you are actually throwing the switch twice. To a toggle-type switch like the soft switch for the speaker, after a "write" operation is completed, the state of the switch remains unchanged.

2.7.2 Cassette Interface

On the back panel (when holding an MPF-II with the keyboard toward you, "top", "bottom", "front", "back", "left", "right" mean the physical top, bottom, front, back, reght, and left of the MPF-II.) of the MPF-II, there are two small holes marked EAR and MIC, respectively. You can plug a cable with a phone plug on each end to your MPF-II and cassette recorder. Then you can store your program from MPF-II to the tape, or load your program from tape to MPF-II.

Note when you load a program from tape to the MPF-II, one end of the cable should plug to EAR of the MPF-II, while the other end of the cable should connect to EAR or OUT on the cassette recorder. If you store a program from the MPF-II to tape, one edn of the cable should plug to MIC of the MPF-II, while the other end of the cable should connect to the jack marked MIC or IN on the cassette recorder.

The connector marked MIC is connected to another soft switch on the MPF-II main board. Like that for the speaker, the soft switch for cassette interface is also a toggle type switch (a two-position switch). The memory location associated with the switch is : 49184 or -16352 (decimal), CØ2ØH (hexadecimal). The program that converts data into recordable tone resides in the monitor program.

The other connector marked EAR is for reading program from tape to the MPF-II. Its major purpose is to decode information stored on tape and to load the decoded data into the MPF-II. Thus, a program stored on tape can be loaded back to the MPF-II for execution.

The input circuit of the MPF-II takes 1V peak-to-peak signal from EARPHONE on the recorder and converts the signal to a string of ones and zeroes. Each time the signal received by the input circuit changes from positive to negative or from negative to positive, the state of the input circuit will be changed. If the input circuit was sending ones, it will start sending zeroes, and vice versa. A program can judge by the value of the memory location 49168 or -16368 or CØ1ØH the state of the input circuit. If the value of the location is greater than or equal to 128, then the state of the input circuit is one. Otherrwise, the stateof the input circuit is zero.

2.7.3 Printer Interface

On the left panel of the MPF-II, there is a connector marked PRINTER. The connector provides printer interface to MPF-II printer or other printers with parallel interface. The pin-out of the printer connector is illustrated in Fig. 2-7.

CHAPTER 3

MONITOR PROGRAM

3.1 Reset

After power up your MPF-II or press the RESET key, your MPF-II will start a reset cycle. The cycle begins by jumping and executing a subroutine in the monitor program. The reset cycle 1) first sets the soft screen switch of video display so the MPF-II will operate in text mode, 2) sets the text window to a full screen size, 3) moves the output cursor to the bottom line of the screen, and 4) sets the character display to normal mode. Those are the series of operations performed by the MPF-II, each time you press the RESET key. In addition, there are two locations in the monitor ROM which are used to determine whether a reset cycle is initialized by powering up the MPF-II or by pressing the RESET key. If a reset cycle is initialized by powering up the MPF-II, it is performing a "cold reset"; otherwise the MPF-II initializes a "warm reset".

3.1.1 Cold Reset (Cold Start)

The cold reset is performed when you turn on your MPF-II. The reset cycle first clears the screen display, displays "MPF-II" on the top line of the screen, and sets two specific memory locations.

3.1.2 Warm Reset (Warm Start)

If you press the RESET key after the MPF-II has already completed a cold start, then the MPF-II will perform a warm start, return to the current language you use, and keep your program and parameters intact.

Resided in the ROM of the MPF-II is the system monitor program, which controls all programs entered and run on the MPF-II and services all programs. Under system monitor conrol, you can

1) Read and change the values in different memory locations.
2) Write your program in machine language and run such program on the MPF-II.
3) Save and write programs to or from your MPF-II.
4) You can move the contents of a memory range to a specified area in memory.
5) You can compare the contents of a memory range with the contents of another specified memory range.
6) You can leave the monitor program and enter any other programs.

3 • 2 Enter Monitor Program

The entry point of MPF-II monitor program is at the memory location FF61H (H stands for hexadecimal. We will refer to hexadecimal numbers with the ending H hereafter.) For your convenience (since you may use the address in decimal in BASIC programming), the entry point is 65377 or -159. You can enter the monitor by calling this location, using the CALL -159 instruction.

Once you entered the monitor program, the monitor prompt @ and cursor will appear at the left edge of the screen. Now the monitor is ready to accept a standard input line. After you have typed an input line, the monitor will not respond unless you press the carrage return key which is marked ↵ (located at the lower right corner on the MPF-II keyboard.) Each input line to the monitor may consist of up to 255 characters. If you want to terminate your stay in the monitor, you can stroke the CONTROL, C, and carriage return. One other way to do this is to press the RESET key.

3.2.1 Communicate with the Monitor

To communicate with the monitor, you must give three types of information to it: command, address, and data. Data and addresses should be given in hexadecimal to the monitor. The decimal numerals from Ø through 9 are used to represent hexadecimal numerals from Ø through 9, and the letters A, B, C, D, E, F are used to represent the decimal numbers from 1Ø to 15. Thus, a hexadecimal digit can be used to represent any of the 16 decimal numbers from Ø to 15, and two hex digits can be used to represent any of the 256 decimal numbers from Ø to 255, and four hex digits can be used to represent 65536 decimal numbers from Ø to 65535. In the MPF-II, every memory location is numbered with four hex digits, and the contents of each memory location are represented by two hex digits.

When the monitor looks for an address, it is actually looking for a group of four hex digits. If a hex digit group (an address) has less than four digits, the monitor assumes the hex digit group comes with leading zero(s). If an address contains more than four hex digits, the monitor will truncate the number and only receive the last four hex digits.

The monitor recognizes 17 different command characters. Some of them are used for punctuation, and the rest (which are upper-case letters) are for issuing commands to the monitor. The first letter of each monitor command is used as a control character. Since the screen does not echo the control characters, be sure to use the control characters carefully. The knowledge of two special memory locations -- the last opened location and the next changeable location -- is necessary to communicate with the monitor properly. The last opened location is the location you last accessed. The next changeable location is the location you may access immediately. Frequently, the last opened location is identical with the next changeable location. You would learn more about the two locations in the course of familiarizing with the monotir commands.

3.2.2 Read the contents of memory

1) Read the contents of a memory location

Type in an address on the input line, the monitor will respond by displaying the address you just typed and the contents of this address. For example,

```
@F800

F800- 4C
@F900

F900- F8
@
```

Every time the monitor display the contents of an address, it remembers that location as the the last opened location. It also regards the location as the next changeable location.

2) Read the contents of a memory block

Typing a period (.) and an address will cause the monitor to perform a memory dump, dumping the contents from the last opened location to the location you typed following the period.

```
@FA00

FA00- D8
@.FA0F

FA01- 62 5A 48 26 62 94 88
FA08- 54 44 C8 54 68 44 E8 94
@F800

F800- 4C
@.F812

F801- CA EF 4A 08 20 F5 FB
F808- 20 73 FC A5 09 28 90 02
F810- 69 0F 85
@.F82A

F813- 09 60 86 EA EA
F818- EA 20 00 F8 C4 2C B0 11
F820- C8 20 C8 EF 90 F6 69 01
F828- 48 20 00
@
```

While performing a memory dump, note the followings:

a. The dumped data displayed on the first output line following the input line (command line) shows the contents in the locations following the last opened location.
b. Every output line displays contents of no more than eight locations.
c. In addition to the first output line, all other output lines' beginning addresses end either with a Ø or an 8.

When the monitor performs a memory dump, it first displays the contents of the location immediately following the last opened location. Then it displays the contents of the following locations one after the other until it proceeds to the location whose address ends with either a Ø or an 8. Contents whose address ends with Ø or 8 are displayed in a new output line. The monitor will continue dumping contents of memory until proceeding to the location you specified after the period. When the monitor stops memory dumping, the location you specified will be considered by the MPF-II as the last opened address.

If the address you specified is less than the address of the last opened location, then only the contents of the single location immediately following the last opened location are to be displayed.

```
@F200.F22F

F200- 00 1C 22 2A 3A 1A 02 3C
F208- 00 08 14 22 22 3E 22 22
F210- 00 1E 22 22 1E 22 22 1E
F218- 00 1C 22 02 02 02 22 1C
F220- 00 1E 22 22 22 22 22 1E
F228- 00 3E 02 02 1E 02 02 3E
@F900.F910

F900- F8 69 BF 20 ED FD CA D0
F908- EC 20 48 F9 A4 2F A2 06
F910- E0
@
```

A simpler way to look into the contents of a memory range (block) is to type in the address of the last opened location, a period, and the ending address.

3) Dumping memory by pressing the carriage return key: Another simpler to perform a memory dump

By pressing the carriage return key, the MPF-II will perform a memory dump to display an output line beginning with the contents of the last opened location. At the same time, the monitor will remember the address of the last displayed location as the last opened location.

```
@F805

F805- 20
@ ↵
 F5 FB
@ ↵

F808- 20 73 FC A5 09 28 90 02
@F832

F832- A0
@ ↵
 2F D0 02 A0 27
@ ↵

F838- 84 2D A0 27 A9 00 85 30
@
```

3.2.3 Altering the contents of memory

1) Alter the contents of a single memory location

We have mentioned before that the last opened location is usually the next changable location. To change the contents of a location, type in a ":" and the data you intend to write into a location.

```
@420

0420- 00
@:12

@420

0420- 12
@
```

Now the value of the next changable location has been changed to 12. If you are not certain of the last opened location, simply type the address of the location whose contents you intend to change, a colon, and the data you want to write to that location. After you have typed in the new data, the original data was changed.

```
@420:23

@420

0420- 23
@
```

2) Alter the contents in consecutive memory locations

To change the contents in consecutive memory locations, you don't have to press an address, a colon, and a value for each location. The monitor allows you to change the contents of 85 consecutive memory locations at a time. All you have to do is to type in the starting address of the memory range, a colon, and a series of data separated by spaces. If you omit the starting address, the MPF-II assumes that the next changable location is the starting address.

```
@400:12 23 34 45 56 67 78 89

@400

0400- 12
@↵
 23 34 45 56 67 78 89
@420:0 1 2 3

@:4 5 6 7

@420.427

0420- 00 01 02 03 04 05 06 07
@
```

4) Move a memory range to a new location

The MOVE command allows you to move a whole memory range to a new location. To complete this task, you have to tell the MPF-II:

a. The destination where you wish to move the memory range to.
b. The starting address of the memory range.
c. The last address of the memory range.

The memory range can be specified the same as before. The first letter of the MOVE command "M" should follow the memory range in upper case. The "<" mark is used to point to the destination. To move a memory range, you should first tell the monitor the destination by typing the address of the destination, then following by the starting address of the memory location, a period, the last address, and finally the upper case command M. The format of the command looks like

```
[Destination] < [Starting address] . [Last address]
```

```
@460.46F

0460- 00 00 00 00 00 00 00 00
0468- 00 00 00 00 00 00 00 00
@400:00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF

@400.40F

0400- 00 11 22 33 44 55 66 77
0408- 88 99 AA BB CC DD EE FF
@460<400.40FM

@460.46F

0460- 00 11 22 33 44 55 66 77
0468- 88 99 AA BB CC DD EE FF
@470.477

0470- 00 00 00 00 00 00 00 00
@470<400.407M

@470.477

0470- 00 11 22 33 44 55 66 77
@
```

If the value of the last address is less than the value of the starting address, then the contents of only one memory location will be moved to the destination. If the address of destination is inside the memory range to be moved, then the following may happen:

```
@420.437

0420- 00 00 00 00 00 00 00 00
0428- 00 00 00 00 00 00 00 00
0430- 00 00 00 00 00 00 00 00
@420:00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF

@426<420.42FM

@420.437

0420- 00 11 22 33 44 55 00 11
0428- 22 33 44 55 00 11 22 33
0430- 44 55 00 11 22 33 00 00
@
```

5) Compare the contents of two memory range

You can use the monitor to compare the contents of two memory ranges, using the same command format as you use to move a memory range. The command format is listed as follows:

[Destination] < [Starting address] . [Ending address]V

The upper-case letter "V" is the first letter of the monitor command VERIFY. To instruct the MPF-II to verify the contents of two memory ranges, you have to type

Destination Address: The starting address of a memory range whose contents are to be compared with those of another memory range;

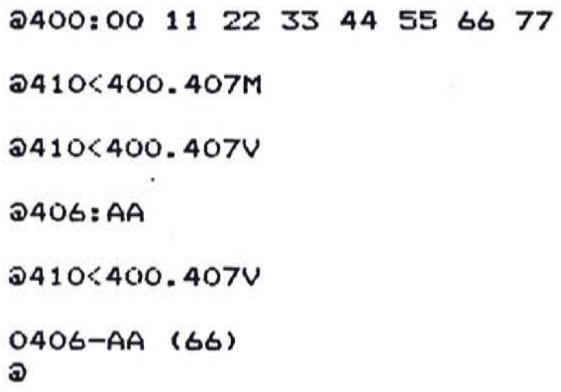
A Left Caret;

The starting address of the memory range you want to compare its contents with those contained in the destination memory range;

A period;

The ending address of the memory range you want to compare its contents with those contained in the destination memory range;

The capital letter "V".



```
@400:00 11 22 33 44 55 66 77

@410<400.407M

@410<400.407V

@406:AA

@410<400.407V

0406-AA (66)
@
```

In case the monitor finds discrepancy, it will display the address where discrepant value was found and the disagreeing values. If there is no discrepancy, nothing will be displayed. Note when discrepancy happens, the displayed address is an address in the original memory range, which you specified following the left caret.

6) Save the contents of a memory range on tape

The WRITE monitor command is used to save the contents of up to 65536 memory locations on cassette tape. You have to tell the monitor the beginning and last addresses of a memory range. The command formats are shown below

a. [Starting address].[Last address]W[filename]

b. [Starting address].[Last address]WA

The capital letter "W" is the first letter of the WRITE command.

The capital letter of "A" is used to tell the monitor that the format of the tape you want to write to is compatible to APPLE II formated tape.

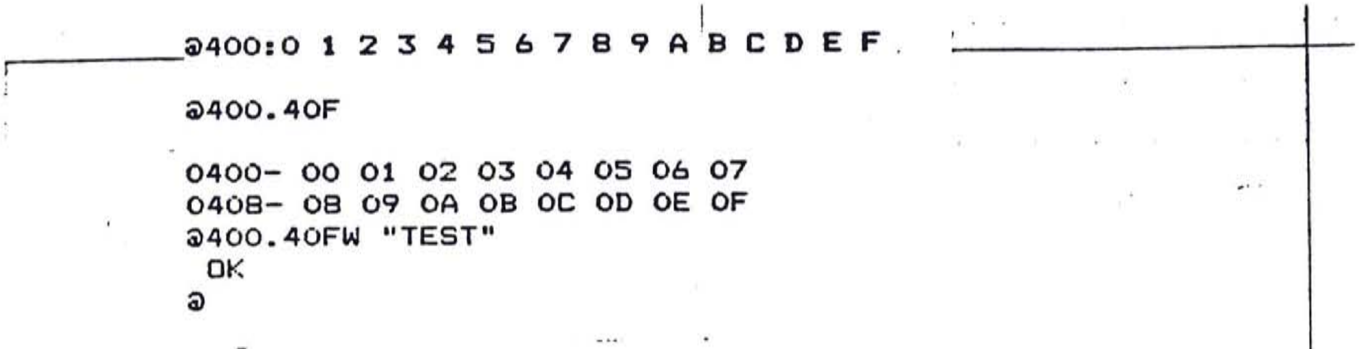
The first WRITE command format is used when writing data into MPF-II formated tapes. A filename may be specified by typing in no more than 6 alphanumeric letters.

The second WRITE format is especially designed for writing data into APPLE II formatted tapes.

In order to allow a recorder to save data correçtly, you have to set it to record mode before typing the carriage return key. Let the tape recorder run a few seconds before pressing the carriage return key. The monitor will first write a "leader tone" unto the tape, and then write data to the tape.

It takes one second for the MPF-II monitor to write a leader tone to an MPF-II formatted tape, and about 1Ø seconds to write a leader tone to APPLE II formatted tape.

After the monitor has saved the data of a memory range on a tape, it will generate a "beep" and display another prompt. A good habit is to rewind the tape and label the the starting and ending addresses of the memory range you just saved on the tape.



```
@400:0 1 2 3 4 5 6 7 8 9 A B C D E F

@400.40F

0400- 00 01 02 03 04 05 06 07
0408- 08 09 0A 0B 0C 0D 0E 0F
@400.40FW "TEST"
 OK
@
```

In addition to the leader tone, it takes 35 seconds to write the contents of 4,Ø96 memory locations to a tape. The monitor will write a checksum on the tape after it has written all the contents of the memory range. The checksum is a partial sum of all the values of the memory range. When the MPF-II reads back a stored a tape in the future,it will determine if the reading is performed without mistake by examing the checksum.

7) Read back a memory range from tape

You can use the READ command to read back the memory range stored on tape with the WRITE command. Remember that the contents of a memory range must be read back to the same memory locations from which they were stored previously. The monitor will put the contents of a memory range to a RAM area in the MPF-II, whose size is exactly the same as that occupied by the memory range stored on tape. The READ commands are listed as follows:

a. [Sarting address].[Last address]R[Filename]

b. [Starting address].[Last address]RA

The first command format is used to find the specified filename in the tape, and then put the file in the RAM of MPF-II specified by the starting address and the last address.

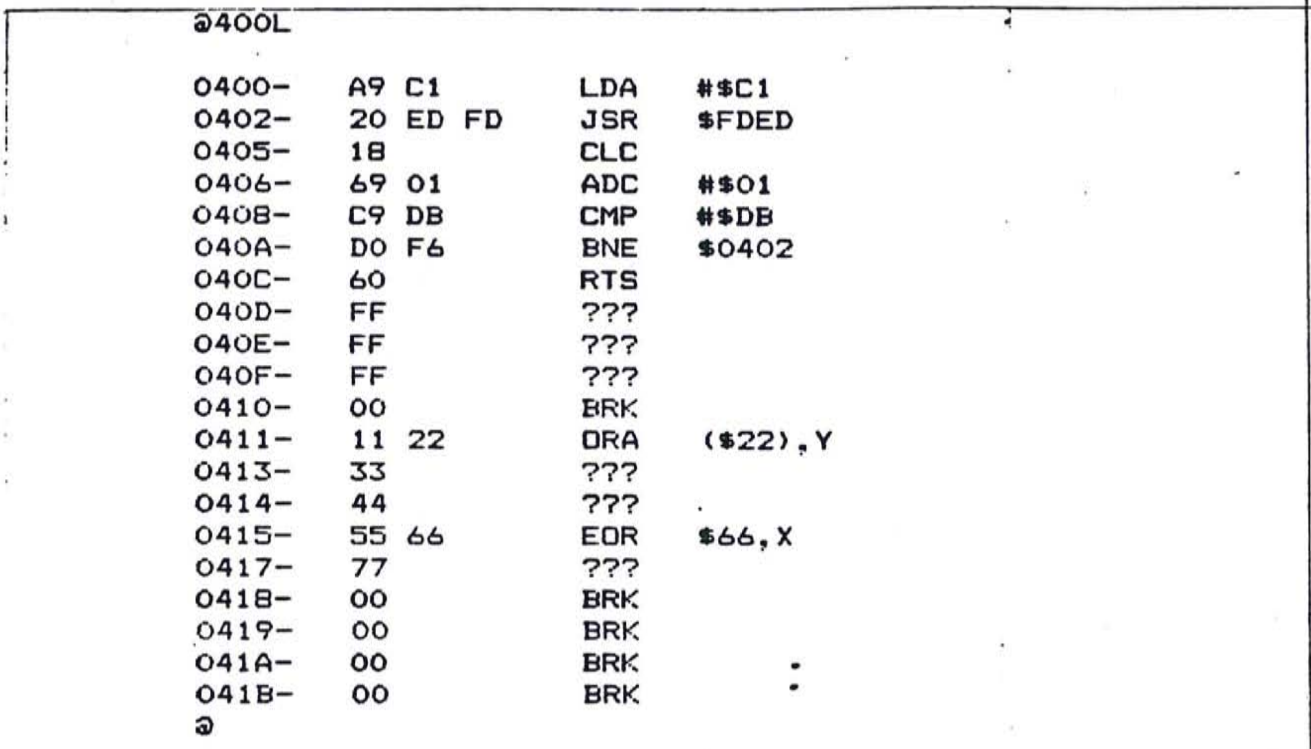
The second format reads data from APPLE II formatted tapes and put the data in a memory range specified by the given starting address and the ending address.

If you write a file onto tape using MPF-II command format, then you have to use the same command format to read data values back into the MPF-II. If you write a file onto tape using the APPLE II compatible command format, you have to use the same command format to read data values back from tape to the MPF-II.

Remember that after you have typed in monitor commands, don't press the carriage return key too hastily. If your tape is of MPF-II format, pressing carriage return and then the PLAY key on the tape recorder will cause the monitor to read the data values stored on tape. If your tape is of APPLE II format, you have to press the PLAY key on the recorder. You will first hear the leader tone. Because the MPF-II needs about three seconds to lock on to the frequency, you shall wait for several seconds to let the tape pass by before pressing the carriage return key.

```
@400:0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

@400.40F

0400- 00 00 00 00 00 00 00 00
0408- 00 00 00 00 00 00 00 00
@400.40FR "TEST"

 TEST   01 01   WAIT.   OK

@400.40F

0400- 00 01 02 03 04 05 06 07
0408- 08 09 0A 0B 0C 0D 0E 0F
@
```

After the monitor has finished reading data values from tape, it will produce a checksum. The monitor will compare this checksum with that it just read in from tape. If the two match, it will produce a prompt on the screen. If the two disagree, the ERR message will display on the screen.



```
@400L

0400-   A9 C1       LDA   #$C1
0402-   20 ED FD    JSR   $FDED
0405-   18          CLC
0406-   69 01       ADC   #$01
0408-   C9 DB       CMP   #$DB
040A-   D0 F6       BNE   $0402
040C-   60          RTS
040D-   FF          ???
040E-   FF          ???
040F-   FF          ???
0410-   00          BRK
0411-   11 22       ORA   ($22),Y
0413-   33          ???
0414-   44          ???
0415-   55 66       EOR   $66,X
0417-   77          ???
0418-   00          BRK
0419-   00          BRK
041A-   00          BRK
041B-   00          BRK
@
```

The lines above are the program we discussed a moment ago. The lines with ??? is just there to fill up the screen.

3 • 4 Other Monitor Commands

You can set the contents of the two memory locations, which is used by the COUT subroutine (Please refer to a detailed explantion of the COUT subroutine in section 3.6.) to switch the video display to either Inverse or Normal mode. The monitor commands NORMAL and INVERSE are used to set the video display mode.

```
@F800.F80F

F800- 4C CA EF 4A 08 20 F5 FB
F808- 20 73 FC A5 09 28 90 02
@I

@F800.F80F

F800- 4C CA EF 4A 08 20 F5 FB
F808- 20 73 FC A5 09 28 90 02
@N

@F800.F80F

F800- 4C CA EF 4A 08 20 F5 FB
F808- 20 73 FC A5 09 28 90 02
@
```

Type in the above program and see how it works.

Pressing CONTROL , C , and the carriage return key allows you to return to the language you were using before entering the monitor. If you were in BASIC before entering the monitor program, the pressing of the three keys lets you return to BASIC. The system monitor can also perform some simple hexadecimal addition and subtraction. You can do some examples by following the procedures listed below

[Value] + [Value]

[Value] - [Value]

[Value] represents a hexadecimal number. See the following examples:

```
@10+12
=22
@3A-B
=2F
@EE+3
=F1
@5-8
=FD
@
```

3 • 5 Some Interesting Features of the Monitor

We have mentioned that the MOVE command can be used to copy the data valaues in a memory range into another range of memory. It can fill up a memory range with the data values we set. Therefore, we can store a few data values into a memory range.

```
@5ØØ : 11 22 33 44
```

Remember we have entered four data values into a memory range. Then you are requested to give a special command to the monitor

```
[ADDR1+NUMBER]<[ADDR1].[ADDR2-NUMBER]M
```

Here NUMBER represents the number of values. In our case now, the number of values is four. The special MOVE command you just entered will duplicate the four values as a set pattern and fill the pattern throughout the memory range specified.

```
@400:11 22 33 44

@404<400.41FM

@400.41F

0400- 11 22 33 44 11 22 33 44
0408- 11 22 33 44 11 22 33 44
0410- 11 22 33 44 11 22 33 44
0418- 11 22 33 44 11 22 33 44
@
```

You can play the same trick with the VERIFY command to check whether a pattern of values has been stored in a memory range or if the locations of a memory range contains the same value.

```
@400:0

@401<400.41FM

@401<400.41FV

@404:02

@401<400.41FV

0403-00 (02)
0404-02 (00)
@
```

3•6 Monitor Command Summary

3.6.1 Examine the contents of memory

[ADDR]
Examine the contents of a single memory location.

[ADDR1].[ADDR2]
Examine the contents of the memory range specified by ADDR1 and ADDR2

3.6.2 Alter the contents of memory

[ADDR]:[VAL1] [VAL2] .....
Put the values behind the colon into the memory range starting at ADDR.

: [VAL1] [VAL2].....
Put the values following the order VAL1, VAL2... into the memory range starting at the next changable location.

3.6.3 MOVE and VERIFY

a. [ADDR1]<[ADDR2].[ADDR3]M
Move the contents from ADDR2 through ADDR3 to the memory range with the starting address ADDR1.

b. [ADDR1]<[ADDR2].[ADDR3]V
Verify the contents of the memory range from ADDR2 through ADDR3 with the memory range starting at ADDR1.

3.6.4 WRITE to and READ from tape

a. [ADDR1].[ADDR2]W[FILENAME]
Write filename and the data values from the memory range ADDR1.ADDR2 onto MPF-II formatted tape.

b. [ADDR1].[ADDR2]R[FILENAME]
Locate filename, read the data from tape, and put the data into the memory location ADDR1.ADDR2.
The command applies to tapes saved with MPF-II format command.

c. [ADDR1].[ADDR2]WA
Write the data values in the memory range ADDR1.ADDR2 to tape. (APPLE II compatible format)

3.6.7 Useful Subroutines

You can apply the useful subroutines listed below in your machine language programs. To use these subroutines, all you have to do is to set the values of proper memory locations or registers, and execute a JSR instruction. Then the MPF-II will perform the desired functions and return with 65Ø2 registers set as described.

1. COUT

[Address]: FDEDH
[Function]: It is the standard character output subroutine. The character to be output should be in the accumulator. COUT calls the current character output subroutine whose address is put in locations 36H and 37H, normally COUT1.

2. COUT1

[Address]: FDFØH
[Function]: Its function is to display the character in the accumulator on the screen at the current cursor position and advance the cursor. Before it displays the character, it will reference the locations controlling the setting of the inverse/normal mode. So the character in the accumulator will be displayed in accordance with the video display mode. It also controls three characters--carriage return, line feed, and bell. The COUT1 returns with all registers unchanged.

3. SETINV

[Address]: FE8ØH
[Function]: Sets the display to inverse mode. All characters will be displayed in black on a monochrome background. It will return with the value of register Y set to 7FH. The contents of all other registers remain unchanged.

4. SETNORM

[Address]: FE84H
[Function]: Sets video display to normal mode. All

output characters will appear in white on a dark screen. The value of register Y is set to ØØH. All other registers are kept intact.

5. CROUT

[Address]: FD8EH
[Function]: Generates a RETURN character to the current output device.

6. CROUT1

[Address]: FD8BH
[Function]: Clears the screen from the current cursor position to the edge of the text window, then calls CROUT1.

7. PRBYTE

[Address]: FDDAH
[Function]: Prints a hexadecimal byte. The subroutine outputs the contents of the accumulator as a hexadecimal byte to the current output device. The contents of the accumulator are changed after execution of this subroutine.

8. PRHEX

[Address]: FDE3H
[Function]: Prints a hex digit. PRHEX outputs the low order nibble of the accumulator as a hex digit. The contents of the accumulator are changed after execuion of this subroutine.

9. PRNTAX

[Address]: F941H
[Function]: Prints the contents of A and X registers in hexadecimal. This outputs the contents of the A and X as a four-digit hexadecimal value. The contents of the accumulator are changed.

1Ø. PRBLNK

[Address]: F948H
[Function]: Prints three spaces. Outputs three spaces to standard output device. The value of A is usually altered to AØH and that of X contains one zero.

11. PRBL2

[Address]: F94AH

[Function]: Prints a series of spaces. Outputs from 1 to 256 spaces to standard output devices. The number of spaces to be output should be stored in X. If X=ØH, then 256 spaces are to be output.

12. BELL

[Address]: FF3AH
[Function]: Outputs a BELL character (CTRL G) to the current output device. Upon exit, the accumulator contains 87H.

13. BELL1

[Address]: FBD9H
[Function]: Causes the MPF-II's speaker to beep at 1 KHz for Ø.1 second. The subroutine alters the contents of A and X.

14. RDKEY

[Address]: FDØCH
[Function]: Fetches an input character. This is the standard character input subroutine. It puts the input cursor on the screen to the position of the current output cursor and jumps to the current input subroutine whose address is stored in locations 38H and 39H, usually KEYIN subroutine.

15. GETLN

[Address]: FD6AH
[Function]: Gets an input line with prompt. It collects input lines. Your machine language programs can use GETLN with the proper prompt character in location 33H. It returns with the input line in the input buffer and X containing the value which represents the length of the input line.

16. GETLNZ

[Address]: FD67H
[Function]: Gets an input line. It is an alternate

entry point for GETLN which generates a carriage return to the standard output before entering into GETLN.

17. GETLN1

[Address]: FD6FH
[Function]: Gets an input line without prompt. GETLN1 is also an alternate entry point for GETLN which does not issue a prompt before it gathers an input line. But if a user deletes a line, GETLN1 will issue the contents of location 33H as a prompt and gets another line.

18. WAIT

[Address]: FCA8H
[Function]: Delays for a specific period of time and then returns to the program which called WAIT. The period of time of the delay is controlled by the value of A. The formula for calculating a time delay is 1/2(26+27A+52AxA)microsecond. A is the value of the accumulator. It returns with the value of A being changed to Ø, but keeps the values of X and Y intact.

19. SETCOL

[Address]: F864H
[Function]: Sets low resolution graphics color. The value of A decides the color to be displayed.

2Ø. NEXTCOL

[Address]: F85FH
[Function]: Increments the value of low resolution color currently in use by 3.

21. PLOT

[Address]: F8ØØH
[Function]: Plot a block on the low resolution graphics screen with preset color. The value of block's vertical coordinate is stored in the accumulator, and the

value of the block's horizontal coordinate is stored is stored in Y register. It returns with the value of the accumulator changed. But the vallues of the X and Y registers remain the same.

22. HLINE

[Address]: F819H
[Function]: Draws a horizontal line of blocks in preset color. The value of vertical coordinate is stored in the accumulator. The value of the horizontal coordinate of the leftmost block is stored in the register Y, and the value of the rightmost block's position is put in memory location 2CH. It returns with the value of the values of A, Y changed. But the value of X remains the same.

23. VLINE

[Address]: F828H
[Function]: Draws a vertical line of blocks in preset color in low resolution graphics. When you call this subroutine, the value of the horizontal coordinate should be stored in Y register, the value of the top vertical coordinate in the accumulator, and the value of the bottom vertical coordinate in 2DH. It returns with the value in A changed.

24. CLRSCR

[Address]: F832H
[Function]: Clears the whole low resolution graphics screen. It returns with the values of the A and Y altered.

25. CLRTOP

[Address]: F836H
[Function]: The function of this subroutine is the same as CLRSCR. The only difference is that CLRTOP simply clears the top 40 lines of the low resolution graphics screen.

26. SCRN

[Address]: F869H
[Function]: This subroutine reads the color of a single block on the low resolution graphics screen. You can call SCRN the same way as calling PLOT. The value of the block it reads is returned in the accumulator. All other registers are not changed.

27. PRERR

[Address]: FF2DH
[Function]: Sends the word "ERR" and a "BELL" character to the standard output device. The contents of the accumulator are changed.

28. IOSAVE

[Address]: FF4AH
[Function]: Saves the contents of all registers in the order of A, X, Y, P, S in locations from Ø7FØH through Ø7F4H. The contents of A and X are changed. The decimal mode is cleared.

29. SCAN1

[Address]: FØ43H
[Function]: Scans the keyboard once. If carry = Ø, no key is pressed. If carry = 1, a key is entered, and the entered key code is stored in the accumulator. Warning! After SCAN1 is executed, the contents in locations 6, 7, 8, 9, 26, 27 on page 1 are changed.

CHAPTER 4

MEMORY STRUCTURE

4 • 1 Introduction

The 65Ø2 microprocessor of the MPF-II can access 65,536 memory locations. You can regard the entire memory of the MPF-II is divided into 256 sections or "pages" with each page containing 256 memory locations. Thus, on page 3Ø, the memory locations range from 3ØØØH to 3ØFFH, totaling 256 locations. Since each address consists of four hexadecimal digits, the first two hex digits (high order byte) can be regarded as the page number, and the low order byte (the last two hex digits) as the location within a page.

The 256 pages of MPF-II memory is divided into three types: 1) RAM, 2) ROM, and 3) input/output (I/O) locations. Different types of memory are used for different purposes. Table 4-1 shows the memory map of the MPF-II.

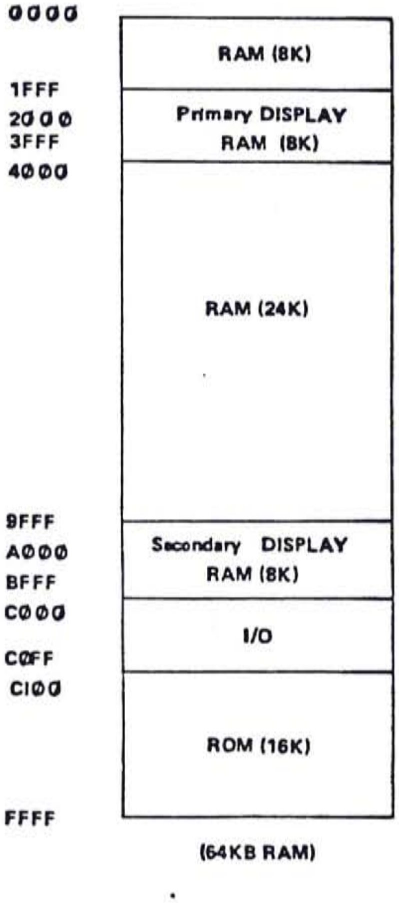


Table 4-1 Memory Map of MPF-II

4 • 2 RAM Area

The RAM area begins from the bottom of page Ø to the end of page 191. This memory range's starting address is ØØØØH and ends at BFFFH. Most of the locations in the RAM is used to store your program and data. But some areas in the RAM are reserved for the monitor, various programming languages, and other system functions. You can refer to Table 4-2 for a description of the uses of the RAM.

Page Number: Decimal	Hex	Used for
Ø	$ØØ	System Programs
1	$Ø1	System Stack
2	$Ø2	GETLN Input Buffer
3	$Ø3	Reserved for Peripheral devices
4	$Ø4	Monitor Program
5	$Ø5	
6	$Ø6	
7	$Ø7	
8	$Ø8	User's RAM
9	$Ø9	
1Ø	$ØA	
11	$ØB	
12	$ØC	
through		
31	$1F	
32	$2Ø	Primary Page for Text, Low-Res, Hi-Res
through		
63	$3F	
64	$4Ø	User's RAM
through		
159	$9F	
160	$AØ	Secondary Page for Text, Low-Res, Hi-Res
through		
191	$BF	

Table 4-2 Description of RAM Usage

The usage of the RAM's various areas is described below:

4.2.1 Zero Page

Nearly 2Ø locations of this page are assigned to the system monitor, while the remaining locations in this page are assigned for use of the MPF-II BASIC. Refer to Table 4-3 and 4-4.

Decimal		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Hex	$0	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
0	$00							●	●	●	●						
16	$10																
32	$20	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
48	$30	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
64	$40	●	●	●	●	●	●	●	●	●	●					●	●
80	$50	●	●	●	●	●	●										
96	$60																
112	$70																
128	$80																
144	$90																
160	$A0																
176	$B0																
192	$C0																
208	$D0																
224	$E0																
240	$F0																

Table 4-3 Monitor Zero Page Usage

Decimal		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Hex	$0	$1	$2	$3	$4	$5	$6	$7	$8	$9	$A	$B	$C	$D	$E	$F
0	$00	●	●	●	●	●	●					●	●	●	●	●	●
16	$10	●	●	●	●	●	●	●	●	●							
32	$20																
48	$30																
64	$40																
80	$50	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
96	$60	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
112	$70	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
128	$80	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
144	$90	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
160	$A0	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
176	$B0	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
192	$C0	●	●	●	●	●	●	●	●	●	●	●	●	●	●		
208	$D0	●	●	●	●	●	●			●	●	●	●	●	●	●	●
224	$E0	●	●	●		●	●	●	●	●	●	●					
240	$F0	●	●	●	●	●	●	●	●	●							

Table 4-4 MPF-II BASIC Zero Page Usage

4.2.2 Page One

The 65Ø2 microprocessor of the MPF-II uses the 256-byte locations of page 1 as a stack. Therefore, make sure your program and data are not stored in this area.

4.2.3 Page Two

The locations of this page are reserved for the GETLN subroutine as a memory buffer for an input line of characters.

4.2.4 Page Three

The locations of page three are reserved for use of peripheral devices to be added in the future.

4.2.5 Page Four through Page Seven

The 1Ø24-byte locations of this area are reserved for the monitor.

4 • 3 ROM Area

The MPF-II is built with 16K ROM, which can hold 16,384 bytes of data. The ROM is mainly used for system monitor, the BASIC Interpreter, utility programs, and subroutines.

The ROM occupies the top 16K locations of the MPF-II's memory map, beginning at location CØØØH and ending at FFFFH. Once the MPF-II is turned on, the 65Ø2 microprocessor jumps to the top of the memory map and begins executing programs. Thus, the programs which begins at the top of the memory map are responsible for initializing the entire system.

Table 4-5 shows the memory map of the ROM of the MPF-II, the programs and subroutines in the ROM.

Cϕϕϕ	CϕFF	I/O
C1ϕϕ	C1FF	FLOPPY DISK BOOTSTRAP
C2ϕϕ	C2CF	PRINTER SOFTWARE DRIVER
C2Dϕ	ECϕ9	BASIC
ECϕA	FFFF	MONITOR ROM

Table 4-5

4 • 4 Input/Output Locations

A total of 256 locations in the memory map of MPF-II are dedicated to input and output functions. This range of memory begins at location CØØØH (49152 or -16384 in decimal) and extends up through location CØFFH (494Ø7 or -16129). The I/O locations ranging from CØØØH to CØ7FH are used by the MPF-II main board, and the remaining are reserved for external use.

CHAPTER 5

INPUT/OUTPUT

Almost all programs and programming languages requires certain types of input from the keyboard, and displays some information (output) on the screen. There is a set of subroutines resided in the MPF-II's ROM, controlling the standard input and output of information.

For the convenience of users to call them, those subroutines are identified by various names.

5 • 1 Standard Output

The standard output subroutine is called COUT (character output), which is used to display upper-case letters, numbers, and special symbols in normal or inverse mode on the screen. In addition to the carriage return, BELL, and backspace, COUT ignores other control characters.

COUT subroutine has its own invisible "output cursor" which points to the position at which the next output character is to be placed. Each time COUT is called, it will put an output character at the current cursor position, replacing the character previously displayed, and moves the cursor one column to the left. If the cursor is moving off the last column of a line (the left edge of the screen), COUT will move the cursor to the first postion on the next line. If the cursor passes the last position of the last line, the COUT scrolls up one line and put the cursor to the first column of the new blank line.

When COUT receives a carriage return character, it will move the cursor to the first location on the next line.

5.1.1 Stop-List Feature

When your program or language sends a carriage return code to COUT, it will examine the keyboard. If you press the CONTROL S key, COUT will stop functioning until you press CONTROL Q. This feature is useful when you list the instructions (statements) of your program. By pressing CONTROL S, the listing of a program will suspend. CONTROL Q will continue the listing of a program.

Anytime pressing the RESET key will suspend the listing or execution of a program.

A backspace character causes COUT to move the invisible cursor one space to the right. If the cursor is at the first position of a line, it remains there even a backspace character was sent to COUT.

The BELL character (CONTROL G) does not cause the COUT to change the display on the screen. However, it causes the MPF-II to generate a tone, whose frequency is 1ØØHz and lasts for one tenth of a second.

5.1.2 Adjusting the text window of the screen

In our discussions of the motions of the output cursor, the words "right", "left", "top", "bottom" are used to describe the physical movements of the sursor on a standard screen 4Ø-column wide and 24-character high. If you don't want a full screen which can display 96Ø characters, you can change the screen display size with the method discussed later. The segregated portion of a screen, whose size is smaller than a full-size screen and which is used for text display, is called a "window". Four memory locations are used to tell COUT the four values that decide the size and position of the text window. After the four values are stored in the four memory locations, COUT will display output characters and move the cursor within the window. The four special memory locations are:

1) Location 32 (2ØH) is assigned to hold the column position of the leftmost column of the window. If the value of this position has never been changed, the default value is normally Ø. The value of this location should never be more than 39 (27H).

2) Location 33 (21H) is assigned to hold the column position of the rightmost column of the window. If the value of this position has never been changed, the initial value is normally 4Ø. When setting the value of this location, be sure that the sum of the two values stored in location 33 and 32 should never exceed 4Ø. In case the sum of the two values exceeds 4Ø, your program and data may be seriously damaged.

3) Location 34 (22H) holds the number of the top line of the window. The value of this location is normally Ø.

4) Location 35 (23H) holds the number of the lines of the desired text window plus one, 24 (16H). The value contained in location 35 can never exceed 24.

Table 5-1 : Text Window Special Locations

Function:	Location:		Minimum/Normal/Maximum Value	
	Decimal	Hex	Decimal	Hex
Left Edge	32	$20	0/0/39	$0/$0/$17
Width	33	$21	0/40/40	$0/$28/$28
Top Edge	34	$22	0/0/24	$0/$0/$18
Bottom Edge	35	$23	0/24/24	$0/$18/$18

5.1.3 Setting display mode: Normal or Inverse

When COUT outputs a character, how will the character be displayed? In normal or inverse? This requires you to look into the contents of the location 5Ø (32H). If the valaue of location 5Ø is ØØ (ØØH), COUT will displays characters in normal mode. If the value of locaion 5Ø is 127 (7FH), the output character will appear in inverse mode. In addition to the two values mentioned above, other values may cause COUT to malfunction.

Table 5-2 Control Values for Inverse/Normal Mode

Value		Effect
Hex	Decimal	
ØØ	ØØH	Display characters in normal mode
127	7FH	Display characters in inverse mode

Everytime a character is to be displayed, a logical "AND" operation is performed between the bits contained in location 5Ø and the outgoing character code. The results of the AND operation is stored in the screen buffer. Thus, the characters to be displayed in inverse mode have character codes range from Ø to 127.

5 • 2 Standard Input

Two subroutines are dedicated to perform the standard input. They are:

* RDKEY (read key): Read a key press from keyboard.
* GETLN (get a line): Get one input line of keystrokes.

5.2.1 RDKEY

The major task of RDKEY is to wait for a key press and report the key code of the pressed key to the program which calls the RDKEY. While RDKEY does this, it also performs two other tasks.

1) Prompting Input:
Once RDKEY is called, it first displays the invisible output cursor on the screen. This accomplishes two things: It reminds the user that the MPF-II is waiting for input. And it indicates to the user where the input information is to be placed on the screen. Usually, the input prompt cursor follows a phrace or a word telling the user what information is now being requested. Since the input cursor is a representation of whatever it was at the position of the invisible output cursor, and the position is usually occupied by a blank space, the input cursor usually appears as a blank square.

   When the user presses a key, RDKEY returns the key code to the program that called it and removes the input cursor. Note that the output cursor represents a position on the screen, but the input cursor is a blank character on the screen. They are usually at the same position and rarely separated from each other, but when the input cursor disappear, the output cursor is still there.

2) Random Number Generating:
While RDKEY waits for the user to press a key, it continually adds 1 to the values of a pair of locations in memory. After a key is pressed, the sum of the values in the two memory locations is set to a number within the range of from Ø to 65,535. The exact sum of the two values is quite unpredictable. Many programs and programming languages use this number as the base of a random number generator. The two memory locations involved are 4EH (78) and 4FH (79).

5.2.2 GETLN

GETLN is the subroutine used to request an input line from the keyboard. After the subroutine gets one line, it returns to the program which called it.

Every time GETLN is called, it first prints a prompt (prompting character). The promptinc character tells you which program has called GETLN, prompting you to enter a line. If it is the monitor, an "@" will appear as a prompt. If it is MPF-II BASIC, the prompt ">" will appear.

From the user's point of view, the MPF-II simply prints a prompt and a cursor. As you type, the characters you type are displayed on the screen and the cursor moves one space to the right each time you stroke a key until you press the carriage return key. After the carriage return key is pressed, GETLN will send the whole input line to your program or the language you are using.

The sequence of actions is: After a prompt appears, GETLN calls RDKEY, which then displays an input cursor. Every time RDKEY returns a key code, GETLN stores the key code in the input buffer and places the corresponding character on where the input cursor was on the screen. The GETLN keeps calling RDKEY until a carriage return key is entered. When GETLN receives a carriage return code, it places this code at the end of the input buffer, clears the remainder of the screen line the input cursor was on, and sends a carriage return code to COUT. GETLN then returns to the program which calls it.

You can press CONTROL X to cancel an entire line, while typing an input line. GETLN will erase the whole line, print a backslash "\", jump to a new line, and print a prompt, enabling you to type in a new input line.

GETLN can receive an input line of up to 255 characters. If your input line exceeds this limit, it will cancel the entire line, and you must retype the line. Thus, GETLN will generate a sound as you proceed to the 249th character.

GETLN allows you to revise and edit the line you are typing so you can make corrections. Listed below is the editing features of GETLN, using the two arrow keys: ←, →.

1) Backspace (←): Pressing backspace key once causes GETLN to erase the one character to the left of the input cursor and send a backspace character to COUT in order to let the input cursor return to the position where a character was erased. If the number of backspace characters sent to COUT is more than the number of characters you have typed, GETLN will erase the entire line.

2) The backspace key can only be used to change the characters on the same input line. In other words, we can not move the cursor to a previous line to make corrections.

3) Retype (→) key: We have mentioned that the backspace key can be used to backspace to a previous position on an input line to modify a character. After you moved back the cursor and made corrections, you may wish to move the cursor leftward to continue typing an input line. In some cases, you may want to skip over several characters of an input line without erasing the characters already typed there. Then retype key is used. Every time the retype key is pressed, the cursor will move one space to the right.

There are two other keys with arrowhead--the line feed key and the up-arrow key. The two keys are designed for playing games. So they do not have editing functions. In other words, you can not move the cusor with the updown arrow keys to change the characters on an input line.

4) Line feed character ↓ : This character causes COUT to move the input cursor one line down, but does not move the cursor horizontally. In case the cursor is on the bottom line on the screen, the screen will scroll up one line.

5) The ↑ key: This key causes the cursor to move up one line, but does not move the cursor horizontally. When the cursor is on the top line on the screen, pressing the ^ key will not change the screen.

5•3 Internal I/O

The input/output functions of MPF-II's main board are controlled by 128 locations in the memory map, extending from CØØØH up to CØ7FH (49152 to 49279 or -16384 to -16257 in decimal). The 128 locations are divided into four types: data output, data input, soft switch, and toggle switch.

5.3.1 Data output

The on-board data output has two functions: an 8-bit data is sent via the data bus to

1) the printer for producing hard copies,
2) the keyboard for keyboard scanning. (Please refer to Appendix C the schematic for keyboard scanning).

5.3.2 Data input

The 8-bit data input to the MPF-II's mainboard is used as:

1) Bit Ø through bit 5 reflect the the current keyboard state. Each time the keyboard is being read, the results are stored in the six bits.
2) Bit 6 reflects the state of the printer.
3) Bit 7 is used to transfer the data input from tape.

5.3.3 Toggle switches

Two strobe outputs are connected to two-state flip-flops. Each time the two locations cooresponding to the strobe are read, the flip-flop will toggle to its other state. The toggle switches are used to drive cassette output and an internal speaker. Because they have only two states, your program can only read from their cooresponding switches, and not write to them. The following example is to cause the speaker to beep continually. First we will enter the monitor program

```
>CALL -159 ←
```

Then you are required to key in the following program:

```
@800:20 06 08 4C 00 08 A9 40 20 19 08 A0
C0 A9 0C 20 19 08 AD 30 C0 88 D0 F5 60
38 48 E9 01 D0 FC 68 E9 01 D0 F6 60 FF

@800L

0800-   20 06 08    JSR   $0806
0803-   4C 00 08    JMP   $0800
0806-   A9 40       LDA   #$40
0808-   20 19 08    JSR   $0819
080B-   A0 C0       LDY   #$C0
080D-   A9 0C       LDA   #$0C
080F-   20 19 08    JSR   $0819
0812-   AD 30 C0    LDA   $C030
0815-   88          DEY
0816-   D0 F5       BNE   $080D
0818-   60          RTS
0819-   38          SEC
081A-   48          PHA
081B-   E9 01       SBC   #$01
081D-   D0 FC       BNE   $081B
081F-   68          PLA
0820-   E9 01       SBC   #$01
0822-   D0 F6       BNE   $081A
0824-   60          RTS
0825-   FF          ???
@800G
```

To stop beeping, you can press the RESET key.

5.3.4 Soft switches

Soft switches are two-position switches. Each side of a soft switch is controlled by a memory location. If you reference the location for one side of a switch, the switch will be set according to the referenced location. If the location for the other side of the switch is referenced, the switch will be set according to the other referenced location. Since the switch is set irrelevant to its former setting, and there is no way to determine the state of a soft switch, you can either read or write the related memory locations to control the soft switches. Table 5-3 is a summary of I/O locations: 5-3

FUNCTION	ADDRESSS	(Read / Write)
DATA OUTPUT	C000	W
DATA INPUT	C010	R
TAPE OUT	C020	R
SPEAKER	C030	R
B / W	C051	R W
COLOR	C050	R W
PRINTER STROBE	C059 HIGH	R W
	C058 LOW	R W
DISPLAY PAGE	C054 (PRIMARY)	R W
	C055 (SECONDARY)	R W
RAM / $\overline{\text{ROM}}$ SELECT	C05A (ROM)	R W
	C05B (RAM)	R W
CONTROL (FOR KEYBOARD USE)	C05E LOW	R W
	C05F HIGH	R W

Table 5-3 Input/output Locations

A summary of Table 5-3:

DATA OUTPUT: When you output 8-bit data to the printer or for keyboard scanning, you have to write a value to the location CØØØH. The value you write to the location is irrelevant, because it is the referencing the address that works.

DATA INPUT: You have to read a value from the location CØ1ØH in case you intend to inform the 65Ø2 CPU: a) the results stored in bit Ø through bit 6--the state of the keyboard, b) if the printer is busy, c) the data input from tape.

TAPE OUT: Read a value from location CØ2ØH in case you want to output data to tape.

SPEAKER: Read a value from location CØ3ØH if you want to generate a tone using the internal speaker.

B/W: Read or write to the location CØ51H for a black and white video display.

COLOR: Read or write to the location CØ5ØH for a color diplay.

PRINTER STROBE: Read or write to the locations CØ58H, CØ59H, CØ58H consecutively to create a strobe pulse. You can also read or write to the locations in the order CØ59H, CØ58H, CØ59H, depending on the pulse you intend to create.

DISPLAY PAGE: Read or write to the two locations controlling the two screen pages for selecting a specific screen page.

$\overline{\text{ROM}}$/RAM SELECT: Read or write to the location CØ5BH will set the memory range C1ØØH through FFFFH as a RAM area. Reference the location CØ5AH will make this memory range a ROM area.

CONTROL: Read the CONTROL keys on the keyboard.

Example 1: Change the video display from black and white to color.

After turning on the MPF-II, you can read from the location CØ5ØH to change the video display from black and white to color.

Example 2: Use two screen pages--primary page and secondary page--to produce alternating screen display.

The I/O location for page 1 or primary page is CØ54H. After the MPF-II is turned on, the primary page is used for screen display. The memory range used for page 1 is from location 2ØØØH and extends up to 3FFFH, totaling 8K bytes.

The secondary page's I/O location is CØ55H. It occupies the memory range starting from location AØØØH up to location BFFFH, totaling 8K bytes. Page 2 is extremely useful for displaying information instantly. You can program the MPF-II to draw cartoon. You can draw a picture on one of the two pages, and another picture on another page. By displaying the picture on one page and then displaying the picture on another page, you can achieve the cartoon effect. Note if your MPF-II only has a RAM of 16K bytes, the primary page is identical to the secondary page.

Example 3: Send a printer strobe.

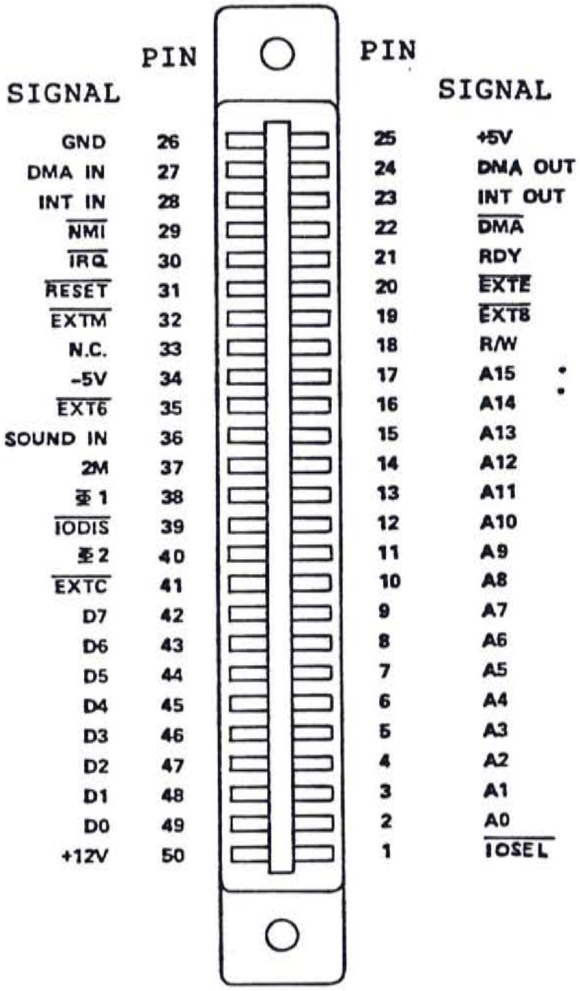
When information is to be sent to a printer, you must first check whether the printer is busy or not. If the printer is not busy, you can produce a ‾|_|‾ pulse by first reading the location CØ59H (high), and then the location CØ58H (low) and location CØ59H (high).If a ‾|_|‾ pulse is to be sent, you should read the locations in the order of CØ58H, CØ59H, CØ58H.

Example 4: Select ROM or RAM.

The memory range from C1ØØH to FFFFH is occupied by the monitor and the BASIC Interpreter. To change this memory range to RAM (you can only do this on MPF-II with 64K byte RAM.), you can read the location CØ5BH.

5 • 4 Peripheral Connector

At the upper left edge of the MPF-II, there is a 5Ø-pin connector used for interfacing with ROM cartridges or other peripheral devices. The pin-out of the edge connector is illustrated in Fig. 5-1



SIGNAL	PIN	PIN	SIGNAL
GND	26	25	+5V
DMA IN	27	24	DMA OUT
INT IN	28	23	INT OUT
$\overline{NMI}$	29	22	$\overline{DMA}$
$\overline{IRQ}$	30	21	RDY
$\overline{RESET}$	31	20	$\overline{EXTE}$
$\overline{EXTM}$	32	19	$\overline{EXT8}$
N.C.	33	18	R/W
-5V	34	17	A15
$\overline{EXT6}$	35	16	A14
SOUND IN	36	15	A13
2M	37	14	A12
Φ1	38	13	A11
$\overline{IODIS}$	39	12	A10
Φ2	40	11	A9
$\overline{EXTC}$	41	10	A8
D7	42	9	A7
D6	43	8	A6
D5	44	7	A5
D4	45	6	A4
D3	46	5	A3
D2	47	4	A2
D1	48	3	A1
D0	49	2	A0
+12V	50	1	$\overline{IOSEL}$

Fig. 5-1 Pin-out of MPF-II Peripheral Connector

Table 5-1 Description of Peripheral Connector Pin-out

Pin number	Name	Description
1	$\overline{\text{IOSEL}}$(OUTPUT)	This pin is normally high. When the microprocessor accesses the memory range from location CØØØH to CØFFH, this line will become low. This line is active only during Φ1.
2-17	AØ - A15 (OUTPUT)	Pin AØ to A15 are buffered address bus. These lines are valid only during Φ1 and remain valid through ΦØ.
18	R/$\overline{\text{W}}$(OUTPUT)	Buffered Read/Write signal. This line is valid only when the address bus is valid. The pin goes high during a read cycle, and is low during a write cycle.
19	$\overline{\text{EXT8}}$	When the microprocessor references the memory location starting from 8ØØØH to 9FFFH, this line becomes low. The signal is active during ΦØ.
2Ø	$\overline{\text{EXTE}}$(OUTPUT)	This line becomes low when the CPU references the memory range from EØØØH to FFFFH. The signal is active during ΦØ.
21	RDY(INPUT)	65Ø2's RDT input. Pulling this line low during Φ1 will halt the 65Ø2 with the address bus holding the address of the location currently being accessed.
22	$\overline{\text{DMA}}$(INPUT)	Pulling this line low will halt the CPU and disable the CPU's address bus. This line is normally held high by a 1K ohm resistor connected to +5V.
23	INT OUT (OUTPUT)	Daisy-chained interrupt output to lower priority devices.
24	DMA OUT (OUTPUT)	DMA output to lower priority devices.

25	+5V	+5V power supply. 2A current is available to all devices.
26	GND	System electrical ground.
27	DMA IN (INPUT)	Daisy-chained DMA input line (connected to higher priority DMA OUT line.
28	INT IN (INPUT)	Daisy-chained interrupt input from higher priority INT OUT line.
29	NMI(INPUT)	Non-maskable interrupt request.
3Ø	IRQ(INPUT)	Interrupt request. When this line is pulled low, MPF-II begins an interrupt cycle only when the 65Ø2's I flag (interrupt disable) is Ø.
31	$\overline{\text{RESET}}$(INPUT)	When this line is pulled low, MPF-II starts a RESET cycle.
32	$\overline{\text{EXTM}}$(INPUT)	When a user accesses the external memory through $\overline{\text{EXTE}}$, $\overline{\text{EXTC}}$, $\overline{\text{EXT8}}$, $\overline{\text{EXT6}}$, the signal is also connected to $\overline{\text{EXTM}}$. When $\overline{\text{EXTM}}$ is pulled low, the memory on the MPF-II main board will not be accessed. Instead, the external memory will be accessed. $\overline{\text{EXTM}}$ is normally connected to +5V through a 1K ohm resister.
33	N.C.	
34	-5V	-5V power supply. The current supplied should be no more than 8ØmA.
35	$\overline{\text{EXT6}}$(OUTPUT)	When the CPU accesses the memory range beginning from 6ØØØH to 7FFFH, this line becomes low. It is active during ΦØ.
36	SOUND IN (INPUT)	Audio signal input. Enable the audio signal from MPF-II to be output from speaker or TV speaker. The peak value should be less than 5V.

37	2M(OUTPUT)	2MHz timing clock pulse output.
38	Φ1(OUTPUT)	65Ø2's phase one clock pulse output.
39	$\overline{\text{IODIS}}$(INPUT)	This line, when pulled low, disables all internal I/O address decoding.
4Ø	Φ2(OUTPUT)	65Ø2's phase two clock.
41	$\overline{\text{EXTC}}$ (OUTPUT)	When the CPU accesses the memory range starting from C1ØØH to DFFFH, this line becomes low. This line is active during ΦØ.
42- 49	DØ - D7 (INPUT/OUTPUT)	Buffered data bus.
5Ø	+12V	+12V power supply. This line supplies up to 125mA for peripheral devices.

Note: The word "OUTPUT" in the above table means that the signal on that pin goes out from the MPF-II, and the word "INPUT" indicates that the signal on that pin is destined for the MPF-II.

APPENDIX A

6502 INSTRUCTION SET

6502 MICROPROCESSOR INSTRUCTIONS

ADC	Add Memory to Accumulator with Carry
AND	"AND" Memory with Accumulator
ASL	Shift Left One Bit (Memory or Accumulator)
BCC	Branch on Carry Clear
BCS	Branch on Carry Set
BEQ	Branch on Result Zero
BIT	Test Bits in Memory with Accumulator
BMI	Branch on Result Minus
BNE	Branch on Result not Zero
BPL	Branch on Result Plus
BRK	Force Break
BVC	Branch on Overflow Clear
BVS	Branch on Overflow Set
CLC	Clear Carry Flag
CLD	Clear Decimal Mode
CLI	Clear Interrupt Disable Bit
CLV	Clear Overflow Flag
CMP	Compare Memory and Accumulator
CPX	Compare Memory and Index X
CPY	Compare Memory and Index Y
DEC	Decrement Memory by One
DEX	Decrement Index X by One
DEY	Decrement Index Y by One
EOR	"Exclusive-Or" Memory with Accumulator
INC	Increment Memory by One
INX	Increment Index X by One
INY	Increment Index Y by One
JMP	Jump to New Location
JSR	Jump to New Location Saving Return Address
LDA	Load Accumulator with Memory
LDX	Load Index X with Memory
LDY	Load Index Y with Memory
LSR	Shift Right one Bit (Memory or Accumulator)
NOP	No Operation
ORA	"OR" Memory with Accumulator
PHA	Push Accumulator on Stack
PHP	Push Processor Status on Stack
PLA	Pull Accumulator from Stack
PLP	Pull Processor Status from Stack
ROL	Rotate One Bit Left (Memory or Accumulator)
ROR	Rotate One Bit Right (Memory or Accumulator)
RTI	Return from Interrupt
RTS	Return from Subroutine
SBC	Subtract Memory from Accumulator with Borrow
SEC	Set Carry Flag
SED	Set Decimal Mode
SEI	Set Interrupt Disable Status
STA	Store Accumulator in Memory
STX	Store Index X in Memory
STY	Store Index Y in Memory
TAX	Transfer Accumulator to Index X
TAY	Transfer Accumulator to Index Y
TSX	Transfer Stack Pointer to Index X
TXA	Transfer Index X to Accumulator
TXS	Transfer Index X to Stack Pointer
TYA	Transfer Index Y to Accumulator

THE FOLLOWING NOTATION APPLIES TO THIS SUMMARY:

A	Accumulator
X, Y	Index Registers
M	Memory
$\overline{C}$	Borrow
P	Processor Status Register
S	Stack Pointer
✓	Change
—	No Change
+	Add
∧	Logical AND
—	Subtract
⊻	Logical Exclusive Or
↑	Transfer From Stack
↓	Transfer To Stack
→	Transfer To
←	Transfer To
V	Logical OR
PC	Program Counter
PCH	Program Counter High
PCL	Program Counter Low
OPER	Operand
#	Immediate Addressing Mode

FIGURE 1 ASL-SHIFT LEFT ONE BIT OPERATION

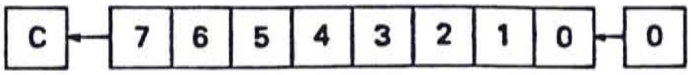


FIGURE 2 ROTATE ONE BIT LEFT (MEMORY OR ACCUMULATOR)

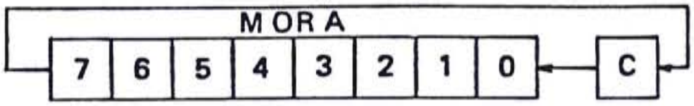
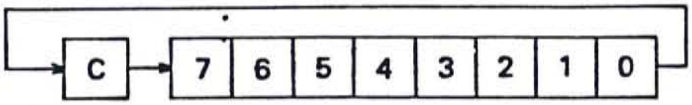


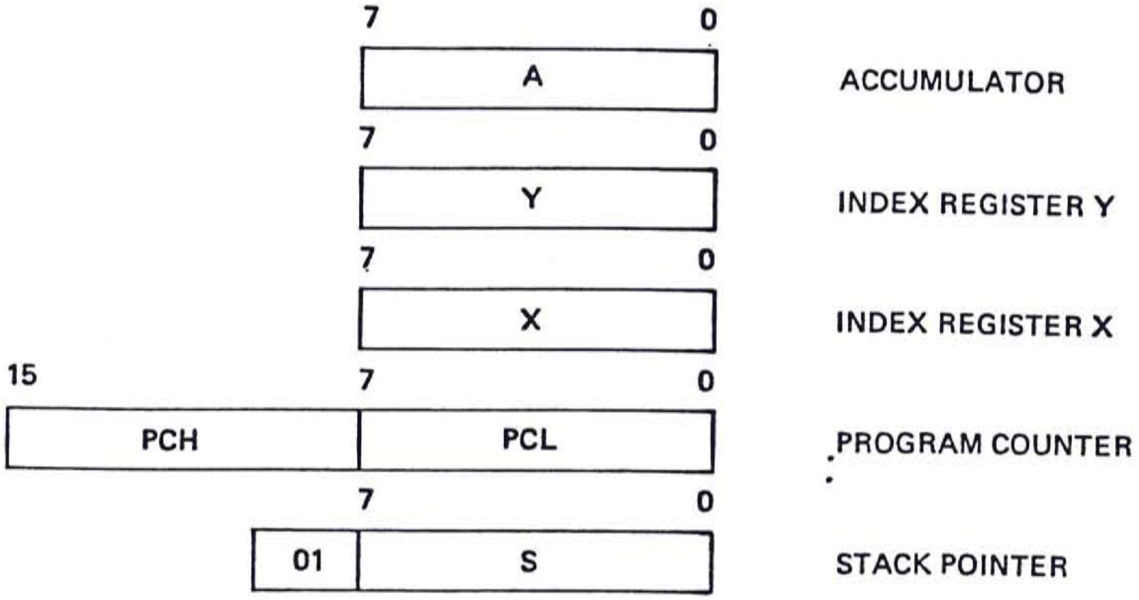
FIGURE 3

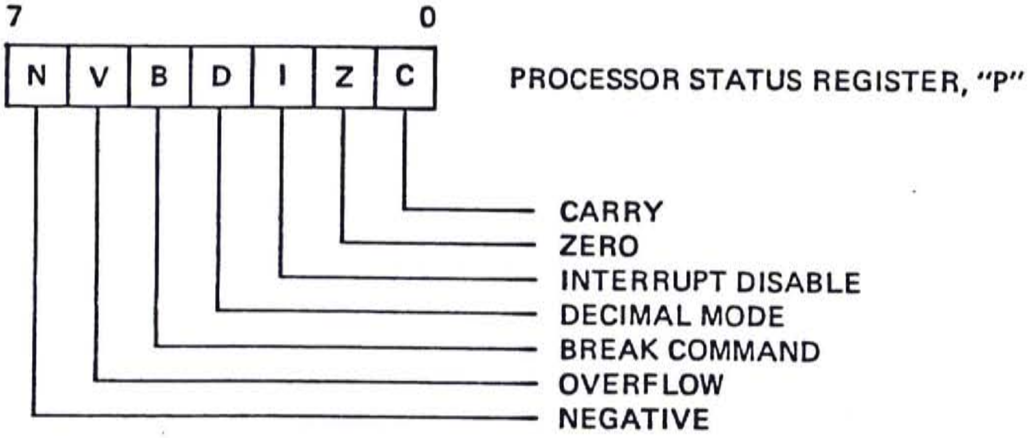


NOTE 1 BIT — TEST BITS

Bit 6 and 7 are transferred to the status register If the result of A ∧ M is zero then Z = 1, otherwise Z = 0

PROGRAMMING MODEL





INSTRUCTION CODES

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. ByTes	"P" Status Reg. N Z C I D V
ADC Add memory to accumulator with carry	A·M·C → A.C	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	ADC #Oper ADC Oper ADC Oper, X ADC Oper ADC Oper, X ADC Oper, Y ADC (Oper, X) ADC (Oper), Y	69 65 75 6D 7D 79 61 71	2 2 2 3 3 3 2 2	✓✓✓——✓
AND "AND" memory with accumulator	A ∧ M → A	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	AND #Oper AND Oper AND Oper, X AND Oper AND Oper, X AND Oper, Y AND (Oper, X) AND (Oper), Y	29 25 35 2D 3D 39 21 31	2 2 2 3 3 3 2 2	✓✓————
ASL Shift left one bit (Memory or Accumulator)	(See Figure 1)	Accumulator Zero Page Zero Page, X Absolute Absolute, X	ASL A ASL Oper ASL Oper, X ASL Oper ASL Oper, X	0A 06 16 0E 1E	1 2 2 3 3	✓✓✓———
BCC Branch on carry clear	Branch on C=0	Relative	BCC Oper	90	2	———————
BCS Branch on carry set	Branch on C=1	Relative	BCS Oper	B0	2	———————
BEQ Branch on result zero	Branch on Z=1	Relative	BEQ Oper	F0	2	———————
BIT Test bits in memory with accumulator	A ∧ M, M_7 → N, M_6 → V	Zero Page Absolute	BIT* Oper BIT* Oper	24 2C	2 3	M_7✓———M_6
BMI Branch on result minus	Branch on N=1	Relative	BMI Oper	30	2	———————
BNE Branch on result not zero	Branch on Z=0	Relative	BNE Oper	D0	2	———————
BPL Branch on result plus	Branch on N=0	Relative	BPL Oper	10	2	———————
BRK Force Break	Forced Interrupt PC+2 ↓ P ↓	Implied	BRK*	00	1	———1——
BVC Branch on overflow clear	Branch on V=0	Relative	BVC Oper	50	2	———————

Note1 B : C and 7 are transferred tothe status register if the resuit of AVM is zero then Z=1 otherwise Z=0
Note2. A BRK command cannot be masked by setting

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
BVS Branch on overflow set	Branch on V=1	Relative	BVS Oper	70	2	——————
CLC Clear carry flag	0 → C	Implied	CLC	18	1	———0——
CLD Clear decimal mode	0 → D	Implied	CLD	D8	1	—0————
CLI	0 → 1	Implied	CLI	58	1	———0——
CLV Clear overflow flag	0 → V	Implied	CLV	B8	1	0—————
CMP Compare memory and accumulator	A – M	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	CMP #Oper CMP Oper CMP Oper, X CMP Oper CMP Oper, X CMP Oper, Y CMP (Oper, X) CMP (Oper), Y	C9 C5 D5 CD DD D9 C1 D1	2 2 2 3 3 3 2 2	✓✓✓———
CPX Compare memory and index X	X – M	Immediate Zero Page Absolute	CPX #Oper CPX Oper CPX Oper	E0 E4 EC	2 2 3	✓✓✓———
CPY Compare memory and index Y	Y – M	Immediate Zero Page Absolute	CPY #Oper CPY Oper CPY Oper	C0 C4 CC	2 2 3	✓✓✓———
DEC Decrement memory by one	M – 1 → M	Zero Page Zero Page, X Absolute Absolute, X	DEC Oper DEC Oper, X DEC Oper DEC Oper, X	C6 D6 CE DE	2 2 3 3	✓✓————
DEX Decrement index X by one	X – 1 → X	Implied	DEX	CA	1	✓✓————
DEY Decrement index Y by one	Y – 1 → Y	Implied	DEY	88	1	✓✓————

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
EDR "Exclusive-Or" memory with accumulator	A V M → A	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	EOR #Oper EOR Oper EOR Oper, X EOR Oper EOR Oper, X EOR Oper, Y EOR (Oper, X) EOR (Oper), Y	49 45 55 4D 5D 59 41 51	2 2 2 3 3 3 2 2	√√————
INC Increment memory by one	M + 1 → M	Zero Page Zero Page, X Absolute Absolute, X	INC Oper INC Oper, X INC Oper INC Oper, X	E6 F6 EE FE	2 2 3 3	√√————
INX Increment index X by one	X + 1 → X	Implied	INX	E8	1	√√————
INY Increment index Y by one	Y + 1 → Y	Implied	INY	C8	1	√√————
JMP Jump to new location	(PC+1) → PCL (PC+2) → PCH	Absolute Indirect	JMP Oper JMP (Oper)	4C 6C	3 3	——————
JSR Jump to new location saving return address	PC+2↓ (PC+1) → PCL (PC+2) → PCH	Absolute	JSR Oper	20	3	——————
LDA Load accumulator with memory	M → A	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	LDA #Oper LDA Oper LDA Oper, X LDA Oper LDA Oper, X LDA Oper, Y LDA (Oper, X) LDA (Oper), Y	A9 A5 B5 AD BD B9 A1 B1	2 2 2 3 3 3 2 2	√√————
LDX Load index X with memory	M → X	Immediate Zero Page Zero Page, Y Absolute Absolute, Y	LDX #Oper LDX Oper LDX Oper, Y LDX Oper LDX Oper, Y	A2 A6 B6 AE BE	2 2 2 3 3	√√————
LDY Load index Y with memory	M → Y	Immediate Zero Page Zero Page, X Absolute Absolute, X	LDY #Oper LDY Oper LDY Oper, X LDY Oper LDY Oper, X	A0 A4 B4 AC BC	2 2 2 3 3	√√————

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
LSR Shift right one bit (memory or accumulator)	(See Figure 1)	Accumulator Zero Page Zero Page, X Absolute Absolute, X	LSR A LSR Oper LSR Oper, X LSR Oper LSR Oper, X	4A 46 56 4E 5E	1 2 2 3 3	0✓✓———
NOP No operation	No Operation	Implied	NOP	EA	1	——————
ORA "OR" memory with accumulator	A V M → A	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	ORA #Oper ORA Oper ORA Oper, X ORA Oper ORA Oper, X ORA Oper, Y ORA (Oper, X) ORA (Oper), Y	09 05 15 0D 1D 19 01 11	2 2 2 3 3 3 2 2	✓✓————
PHA Push accumulator on stack	A ↓	Implied	PHA	48	1	——————
PHP Push processor status on stack	P ↓	Implied	PHP	08	1	——————
PLA Pull accumulator from stack	A ↑	Implied	PLA	68	1	✓✓————
PLP Pull processor status from stack	P ↑	Implied	PLP	28	1	From Stack
ROL Rotate one bit left (memory or accumulator)	(See Figure 2)	Accumulator Zero Page Zero Page, X Absolute Absolute, X	ROL A ROL Oper ROL Oper, X ROL Oper ROL Oper, X	2A 26 36 2E 3E	1 2 2 3 3	✓✓✓———
ROR Rotate one bit right (memory or accumulator)	(See Figure 3)	Accumulator Zero Page Zero Page, X Absolute Absolute, X	ROR A ROR Oper ROR Oper, X ROR Oper ROR Oper, X	6A 66 76 6E 7E	1 2 2 3 3	✓✓✓———

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
RTI Return from interrupt	P ↑ PC ↑	Implied	RTI	40	1	From Stack
RTS Return from subroutine	PC ↑. PC+1 → PC	Implied	RTS	60	1	——————
SBC Subtract memory from accumulator with borrow	$A - M - \overline{C} \rightarrow A$	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	SBC #Oper SBC Oper SBC Oper, X SBC Oper SBC Oper, X SBC Oper, Y SBC (Oper, X) SBC (Oper), Y	E9 E5 F5 ED FD F9 E1 F1	2 2 2 3 3 3 2 2	√√√———
SEC Set carry flag	1 → C	Implied	SEC	38	1	——1———
SED Set decimal mode	1 → D	Implied	SED	F8	1	————1—
SEI Set interrupt disable status	1 → I	Implied	SEI	78	1	———1——
STA Store accumulator in memory	A → M	Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	STA Oper STA Oper, X STA Oper STA Oper, X STA Oper, Y STA (Oper, X) STA (Oper), Y	85 95 8D 9D 99 81 91	2 2 3 3 3 2 2	——————
STX Store index X in memory	X → M	Zero Page Zero Page, Y Absolute	STX Oper STX Oper, Y STX Oper	86 96 8E	2 2 3	——————
STY Store index Y in memory	Y → M	Zero Page Zero Page, X Absolute	STY Oper STY Oper, X STY Oper	84 94 8C	2 2 3	——————
TAX Transfer accumulator to index X	A → X	Implied	TAX	AA	1	√√————
TAY Transfer accumulator to index Y	A → Y	Implied	TAY	A8	1	√√————
TSX Transfer stack pointer to index X	S → X	Implied	TSX	BA	1	√√————

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
TXA Transfer index X to accumulator	X → A	Implied	TXA	8A	1	√√————
TXS Transfer index X to stack pointer	X → S	Implied	TXS	9A	1	——————
TYA Transfer index Y to accumulator	Y → A	Implied	TYA	98	1	√√————

HEX OPERATION CODES

00 — BRK
01 — ORA— (Indirect, X)
02 — NOP
03 — NOP
04 — NOP
05 — ORA— Zero Page
06 — ASL — Zero Page
07 — NOP
08 — PHP
09 — ORA— Immediate
0A — ASL — Accumulator
0B — NOP
0C — NOP
0D — ORA— Absolute
0E — ASL — Absolute
0F — NOP
10 — BPL
11 — ORA— (Indirect), Y
12 — NOP
13 — NOP
14 — NOP
15 — ORA— Zero Page, X
16 — ASL — Zero Page, X
17 — NOP
18 — CLC
19 — ORA— Absolute, Y
1A — NOP
1B — NOP
1C — NOP
1D — ORA— Absolute, X
1E — ASL — Absolute, X
1F — NOP
20 — JSR
21 — AND— (Indirect, X)
22 — NOP
23 — NOP
24 — BIT — Zero Page
25 — AND— Zero Page
26 — ROL— Zero Page
27 — NOP
28 — PLP
29 — AND— Immediate
2A — ROL— Accumulator
2B — NOP
2C — BIT — Absolute
2D — AND— Absolute
2E — ROL— Absolute
2F — NOP
30 — BMI
31 — AND— (Indirect), Y
32 — NOP
33 — NOP
34 — NOP
35 — AND— Zero Page, X
36 — ROL— Zero Page, X
37 — NOP
38 — SEC
39 — AND— Absolute, Y
3A — NOP
3B — NOP
3C — NOP
3D — AND— Absolute, X
3E — ROL— Absolute, X
3F — NOP
40 — RTI
41 — EOR— (Indirect, X)
42 — NOP
43 — NOP
44 — NOP
45 — EOR— Zero Page
46 — LSR — Zero Page
47 — NOP
48 — PHA
49 — EOR— Zero Page
4A — LSR — Accumulator
4B — NOP
4C — JMP — Absolute
4D — EOR— Absolute
4E — LSR — Absolute
4F — NOP
50 — BVC
51 — EOR— (Indirect), Y
52 — NOP
53 — NOP
54 — NOP
55 — EOR— Zero Page, X
56 — LSR — Zero Page, X
57 — NOP
58 — CLI
59 — EOR— Absolute, Y
5A — NOP
5B — NOP
5C — NOP
5D — EOR— Absolute, X
5E — LSR — Absolute, X
5F — NOP
60 — RTS
61 — ADC— (Indirect, X)
62 — NOP
63 — NOP
64 — NOP
65 — ADC— Zero Page
66 — ROR— Zero Page
67 — NOP
68 — PLA
69 — ADC— Immediate
6A — ROR— Accumulator
6B — NOP
6C — JMP — Indirect
6D — ADC— Absolute
6E — ROR— Absolute
6F — NOP
70 — BVS
71 — ADC— (Indirect), Y
72 — NOP
73 — NOP
74 — NOP
75 — ADC— Zero Page, X
76 — ROR— Zero Page, X
77 — NOP
78 — SEI
79 — ADC— Absolute, Y
7A — NOP
7B — NOP
7C — NOP
7D — ADC— Absolute, X NOP
7E — ROR— Absolute, X NOP
7F — NOP
80 — NOP
81 — STA — (Indirect, X)
82 — NOP
83 — NOP
84 — STY — Zero Page
85 — STA — Zero Page
86 — STX — Zero Page
87 — NOP
88 — DEY
89 — NOP
8A — TXA
8B — NOP
8C — STY — Absolute

8D — STA — Absolute
8E — STX — Absolute
8F — NOP
90 — BCC
91 — STA — (Indirect), Y
92 — NOP
93 — NOP
94 — STY — Zero Page, X
95 — STA — Zero Page, X
96 — STX — Zero Page, Y
97 — NOP
98 — TYA
99 — STA — Absolute, Y
9A — TXS
9B — NOP
9C — NOP
9D — STA — Absolute, X
9E — NOP
9F — NOP
A0 — LDY — Immediate
A1 — LDA — (Indirect, X)
A2 — LDX — Immediate
A3 — NOP
A4 — LDY — Zero Page
A5 — LDA — Zero Page
A6 — LDX — Zero Page
A7 — NOP
A8 — TAY
A9 — LDA — Immediate
AA — TAX
AB — NOP
AC — LDY — Absolute
AD — Absolute
AE — LDX — Absolute
AF — NOP
B0 — BCS
B1 — LDA — (Indirect), Y
B2 — NOP
B3 — NOP

B4 — LDY — Zero Page, X
B5 — LDA — Zero Page, X
B6 — LDX — Zero Page, Y
B7 — NOP
B8 — CLV
B9 — LDA — Absolute, Y
BA — TSX
BB — NOP
BC — LDY — Absolute, X
BD — LDA — Absolute, X
BE — LDX — Absolute, Y
BF — NOP
C0 — CPY — Immediate
C1 — CMP — (Indirect, X)
C2 — NOP
C3 — NOP
C4 — CPY — Zero Page
C5 — CMP — Zero Page
C6 — DEC — Zero Page
C7 — NOP
C8 — INY
C9 — CMP — Immediate
CA — DEX
CB — NOP
CC — CPY — Absolute
DC — CMP — Absolute
CE — DEC — Absolute
CF — NOP
D0 — BNE
D1 — CMP — (Indirect), Y
D2 — NOP
D3 — NOP
D4 — NOP
D5 — CMP — Zero Page, X
D6 — DEC — Zero Page, X
D7 — NOP
D8 — CLD
D9 — CMP — Absolute, Y
DA — NOP

DB — NOP
DC — NOP
DD — CMP — Absolute, X
DE — DEC — Absolute, X
DF — NOP
E0 — CPX — Immediate
E1 — SBC — (Indirect, X)
E2 — NOP
E3 — NOP
E4 — CPX — Zero Page
E5 — SBC — Zero Page
E6 — INC — Zero Page
E7 — NOP
E8 — INX
E9 — SBC — Immediate
EA — NOP
EB — NOP
EC — CPX — Absolute
ED — SBC — Absolute
EE — INC — Absolute
EF — NOP
F0 — BEQ
F1 — SBC — (Indirect), Y
F2 — NOP
F3 — NOP
F4 — NOP
F5 — SBC — Zero Page, X
F6 — INC — Zero Page, X
F7 — NOP
F8 — SED
F9 — SBC — Absolute, Y
FA — NOP
FB — NOP
FC — NOP
FD — SBC — Absolute, X
FE — INC — Absolute, X
FF — NOP

APPENDIX B

GLOSSARY

6502: The name of the microprocessor which is the brain of the MPF-II. Manufacturers of this microprocessor include Rockwell, Synertek, MOS Technology.

Address: When used as a noun, it is the specific value given to a memory location. The memory of the MPF-II ranges from Ø to 65535. As a verb, to refer to a specific memory location.

Addressing Mode: The methods with which a microprocessor refers to memory locations. The 65Ø2 has thirteen addressing modes. Thus, it has thirteen ways of referring to memory.

Analog: Using continously variable numbers to represent physical quantities such as voltage, length, resistance, etc. (Contrasted with digital).

AND: A logical operator or binary function that has the property that if P (an input) statement and Q (another input) statement are both true, then P AND Q are true (positive or on), false (negative or off) if either is false or both are false.

ASCII: American Standard Code for Information Interchange (often called USASCII). A standard 8-bit information coding system that assigns a unique value from Ø to 127 to each of 128 characters, numbers, special characters, and control characters. It is used widely with most computers and terminals and can be transmitted parallel or serial.

Assembler: A program used to convert a source assembly language program, including mnemonics and symbols, to an object code (in binary representation) that can be recognized by the computers.

Assembly language: The structure of this programming language is similar to that of machine code language. Programs written in assembly language consist of mnemonics, symbols, and values. It is much more efficient to program in assembly language than to program in other programming languages.

BASIC: Beginner's All-purpose Symbolic Instruction Code. Developed at Dartmouth College by Kemney and Kurtz in 1963, is probably one of the easiest programming languages to learn and master, and is probably the most widely used language for home and personal computers.

Binary: A number system with only two digits "Ø" and "1". Inside all computers, binary arithmetic operations are used for quick computing.

Binary function: An operation performed by an electronic circuit which has one or more inputs and only one output. All inputs and outputs are binary signals.

Bit: The most basic unit of data that can be recognized and processed by all computers. A bit is either Ø or 1. Bits can be grouped to form larger unit of information--nibble, byte.

Board: See Printed circuit board.

Bootstrap (boot): To start a system from a "cold" reset cycle. This term was derived from the sense that the computer intends to get itself started by pulling its bootstraps.

Buffer: An area in the memory in which information is to be stored temporarily, and will be output or processed later.

Bug: An error. A hardware bug is an electrical, mechanical, or electronic defect that interfers with the normal operation of a computer. A software bug is a mistake in programming.

Bus: A group of wires used to carry a set of related signals or information from one place to another.

Byte: A basic unit of information that is stored or processed by the computer. In some computers, a byte consists of seven bits. But in most of today's computers, a byte is made up of eight bits. For a byte consisted of eight bits, one byte can represent or hold a value of from Ø up to 255. Each character in the ASCII character set can be represented by one byte. Each and every memory location of the MPF-II is one 8-bit byte.

Call: As a verb. To leave the program or the subroutine which is now being executed, jump to another program and execute that program, and then return to the original program. As a noun, an instruction that calls a subroutine or program.

Character: One symbol of a set of basic symbols such as those corresponding to the keys on a standard typewriter keyboard. The symbols usually include the decimal letters from Ø to 9, the letters from A through Z, punctuation marks, operation symbols, or other special signs the computer can read, write, and process.

Chip: The equivalent of integrated circuit--tiny silicon chip on which electronic circuits are implemented on a thin metal oxide film.

Code: As a noun. A system of symbols for representing data or instructions.

Cold-start: To begin to operate computer which has just been turned on.

Color burst: The signal that color TV sets recognize and convert to the color dots which are then displayed on the TV screen. Without the color busrt signal, all pictures would be black-and-white.

Computer: Any device that can receive and store a set of instructions and information and process them in a predeermined and predictable way.

Control character: Characters in ASCII character set which have no graphic represenation, and thus can not be seen on the screen. These characters are used for various control functions.

CRT: Cathod ray tube. Usually used to describe any device that has a TV screen.

Cursor: A special symbol on a screen, showing the position where a character can be typed in.

Data: Information of any type.

Debug: Find and correct errors.

DIP: Dual in line package. It is the most popular package method for integrated circuits. It has two parallel rows of pins. The numbers of pins usually come in 14, 16, 18, 2Ø, 24, 4Ø.

Disassembler: A program used to convert object machine code programs to assembly language programs.

Display: As a noun, denotes any device that displays. Usually, the video display is a TV screen.

Edge connector: A connector to which the edge of a printed circuit board can plug for electronic signal exchanging.

Entry point: Specific points or places in a subroutine where control can be transferred and re-entered. The entry point usually corresponds to a new or different function to be performed.

Exclusive OR: A binary function whose result is "off" or "false" when both inputs are "on" (true) or "off" (false); "on" when only one input is "on" (true).

Format: A predetermined ways to arrange characters, lines, punctuation, lines, etc. As a verb, to specify the form of a format.

Graphic: Visiable as a distinct, recognizable shape or color.

Graphics: A capability or a system to display graphic items.

Hardware: The physical components of a computer.

HC: Hard copy. As a verb, to print what is on the screen onto paper, using a printer.

Hexadecimal: A number system in which the decimal Ø through 9 are used to represent the hexadecimal Ø through 9, and the decimal values 1Ø through 15 are represented by the letters from A to F. The hexadecimal values in this manual end with an H.

High level language: The programming languages that are close to human languages.

High order: The most important, or item which contains the highest value, of a set of similar items. The high order bit is the leftest or the most significant bit.

High part: The high order byte of a two-byte address. In decimal, the high part of an address is the quotient of the address divided by 256. In 65Ø2 and many other microprocessors, the high part of an address comes last when stored in memory.

Hz: Hertz, a measurement unit of frequency, cycle per second. 65Ø2 operates at 1,Ø23,ØØØ Hz.

I/O: Input/Output.

Input: As a noun, data which flows from external devices to a computer. As a verb, to send data from external components.

Input/Output: The software or hardware that exchanges data with the external devices.

Instruction: In a program, the smallest unit of data that can be operated upon, processed by a microprocessor (CPU). In 65Ø2 machine language, a instruction may come in one, two, or three bytes.

Integrated circuit: A small, thin silicon wafer into which complicated electronic circuit has been implemented. A single IC can hold from ten to ten thousand discrete electronic components. These ICs are usually packaged in DIP.

Interface: An exchange of information between one device and another, or the device that makes such exchange possible.

Interpreter: A program that converts high level programs to executable machine codes and execute the interpreted machine codes.

Interrupt: A physical effect that causes the computer to jump to and execute an interrupt service subroutine (a subroutine that requests the CPU to halt its current job and service the subroutine that made the request.) When the computer completed servicing the requesting subroutine, control is returned to where the CPU was previously interrupted.

K: Stands for the Greek prefix "kilo", meaning one thousand. In computer usuage, it means 210, or 1,024.

Kilobyte: K bytes, 1024 bytes.

Language: A computer language is a set of characters or a code that is used to form symbols, words, etc., and the rules for combining these into meaningful communications that can be understood by man and the computer.

Line: On a screen, a line is a horizontal string of graphic symbols, extending from one end of the screen to the other. In the MPF-II, an input line is a string of up to 256 characters, terminated by the control character return.

Low level language: Fundamental computer languages whose structure is suitable for the computer to recognize and process. The assembly language is a low level language.

Low order: The least important, or item with the least value in a set of similar items. The low order bit of a byte is the rightmost bit of the byte.

Low part: In a two-byte address, the low order part is the low order byte of the two-byte address.

Machine language: The language that can be understood by the computer. Basically, it comes in binary. Sets of binary values are grouped to form operation codes (opcode) that specify the operations to be performed by the computer. The opcodes operate upon, or sometimes without, operands (represented also by binary values). An instruction may consist of an opcode and an operand, it may also be an opcode.

Memory address: A value assigned to a memory location. An address only corresponds to a single memory location. The address of a memory location may be expressed in hexadecimal or decimal. For the MPF-II, an address is a two-byte value.

Memory location: The smallest unit of data that can be processed by the computer. It is also the smallest unit of information of a memory map to which the computer can refer. A memory location corresponds with a specific address and a unique value.

Memory map: The set of all memory locations which the microprocessor can address directly. It is used as a graphic representation of a computer system's memory.

Microcomputer: A computer whose CPU is a microprocessor.

Microprocessor: An IC that understands and executes machine language programs.

Mnemonic: A set of symbols designed to help memorize data that is difficult to remember. In assembly language, each instruction is represented by a mnemonic.

Mode: A condition or set of conditions under which a set of rules apply.

Monitor: 1) A program that supervises and services all user's programs. It enables a programmer to operate the computer with assembly programs. 2) a close-circuit TV receiver.

Multiplexer: An electronic circuit which has many data inputs, several selector inputs, and one output.

Mux: Multiplexer.

Nibble: Half of a byte, or four bits.

Opcode: A machine language instruction.

Or: A binary function whose output is true or "on" if both or one input is true ("on").

Output: As a noun, data generated by the computer whose destination is the outside world. As a verb, the process of generating such a data.

Page: 1) A unit of measurement for the quantity of memory. A page of memory holds 256 bytes. 2) A full screen of information on a video display.

Pascal: 1) A famous French scientist. 2) A programming language.

PC board: Printed circuit board.

Peripheral: Devices connected to the computer. Most peripherals are input and/or output devices.

Personal computer: Computers with memory, languages, and peripherals which are suitable for use in a home, office, or school.

Pin-out: A description of the function of each pin on an IC.

Printed circuit board: A board of fiberglass or epoxy on which a thin layer of metal has been applied, and then etched away to form traces. Electronic components can be attached to the traces for exchanging signals. Small PC board are usually called cards.

Program: A sequence of instructions created for perform a specific task.

PROM: Programmable Read Only Memory. It is a ROM whose contents can be altered by electrical means. Information stored in PROM remains unchanged when the power is turned off. The contents in PROMs can be erased by ultraviolet light and be reprogrammed.

Prompt: A special symbol which indicates the beginning of an input line.

RAM: Random Access Memory, the memory IC to and from which data can be written and retrieved freely. The contents of RAM will disappear as soon as the power is turned off.

ROM: Read Only Memory, the memory IC in which data is stored permanently and can not be altered. The data in ROM remains intact even if the power is turned off. ROMs are ususally used to store important programs or data such as the monitor program. Data in ROMs is implemented there in the manufacturing process.

Reference: 1) The soùrce of information, such as reference books. 2) As a verb, the action of accessing a memory location.

Return: To exit a subroutine and go back to the program which calls it.

Run: Execute a program in accordance with the sequence of the instrucions in the program.

Scan line: A single beam of a cathode beam across the inner surface of a cathode ray tube.

Schematic: A diagram which illustrates the electrical interconnections and the circuitry of an electronic device.

Scroll: To move all the text on a screen up one line.

Soft switch: A switch, usually a toggle type (on/off) switch, controlled by the software.

Software: Programs that request the hardware to do something.

Stack: An area in memory used to temporarily store data. Data is stored in a stack first-in last-out. The data which is first stored in the stack is retrieved last. A programmer stores (PUSH) data onto the stack, and retrieves (POP) it from the stack.

Strobe: A selection signal which indicates the occurence of a specific event.

Subroutine: A program that can be executed, using the CALL instruction.

Syntax: The rules governing sentence structure in a language.

Text: A collection of characters.

Toggle switch: An on/off switch which has two states--either on or off.

Trace: An etched conducive line on a PC board.

Video: 1) Any information that is displayed on a cathode ray tube. 2) Anything that is visible.

Warm-start: To re-initialize a computer system after a programmer lost control of a program or the operating system.

Window: The segmented portion of a screen is called a window within which information is displayed.

APPENDIX C

SCHEMATIC OF KEYBOARD

FDI-MPF-II

FLOPPY DISK INTERFACE INSTALLATION GUIDE

MULTITECH INDUSTRIAL CORPORATION

FDI-MPF-II

FLOPPY DISK INTERFACE
INSTALLATION GUIDE

TABLE OF CONTENTS

Chapter 1 Unpacking and Installation DOS:file

It is assumed that the reader is already familiar with the Apple II DOS Manual

1. Unpacking the Floppy Disk Interface (FDI)

Welcome to the Multitech floppy disk interface card. Upon unpacking, you will find:

1) The FDI itself.
2) This manual.
3) The diskette containing the following convertor programs:

Filename	Function
	Apple DOS to MPF-II DOS, which has no filename.
FID.COVT	The Apple FID (file developer) to MPF-II FID, whose filename is FID.COVT.
COPY.COVT COPYM.OBJ	The Apple COPYA to MPF-II COPYM, whose filename is COPYA.COVT.

Other programs on the diskette are:

HELLO	The greeting program--HELLO.
O/X GAME	The O/X game.
NURSE	The MPF-II self-diagnosis program--Micro-Nurse.
DEMO-GRAPH	The graphics demonstration program.
DEMO-3ØØ-CCC	The demostration program for Chinese Character Controller (CCC).
RENUMBER	The RENUMBER utility program.
RENUMBER INSTRUCTIONS	The instructions on how to use the RENUMBER utility.
DEMO-SONG-SSG	The demostration program for Speech and Sound Generator (SSG).

4) Two floppy diskettes: One contains the convertor programs and other programs mentioned above and the other is a blank diskette.

2. Installation

Before inserting the FDI to the edge connector of the MPF-II, make sure the power to the MPF-II is turned off. Otherwise, permanent damage may be caused to the FDI.

1) Insert the FDI to the edge connector of the MPF-II

caused to the FDI.

1) Insert the FDI to the edge connector of the MPF-II as illustrated in Fig. 1-1. Note that the side labeled with the floppy disk interface should face up. Otherwise, it can not be plugged into the edge connector of the MPF-II.

2) Insert the flat cable to the FDI as illustrated in Fig. 1-2. Note that the side of the cable connector with a small protrusion should face downwards when making the connection.

3) Turn on the power.

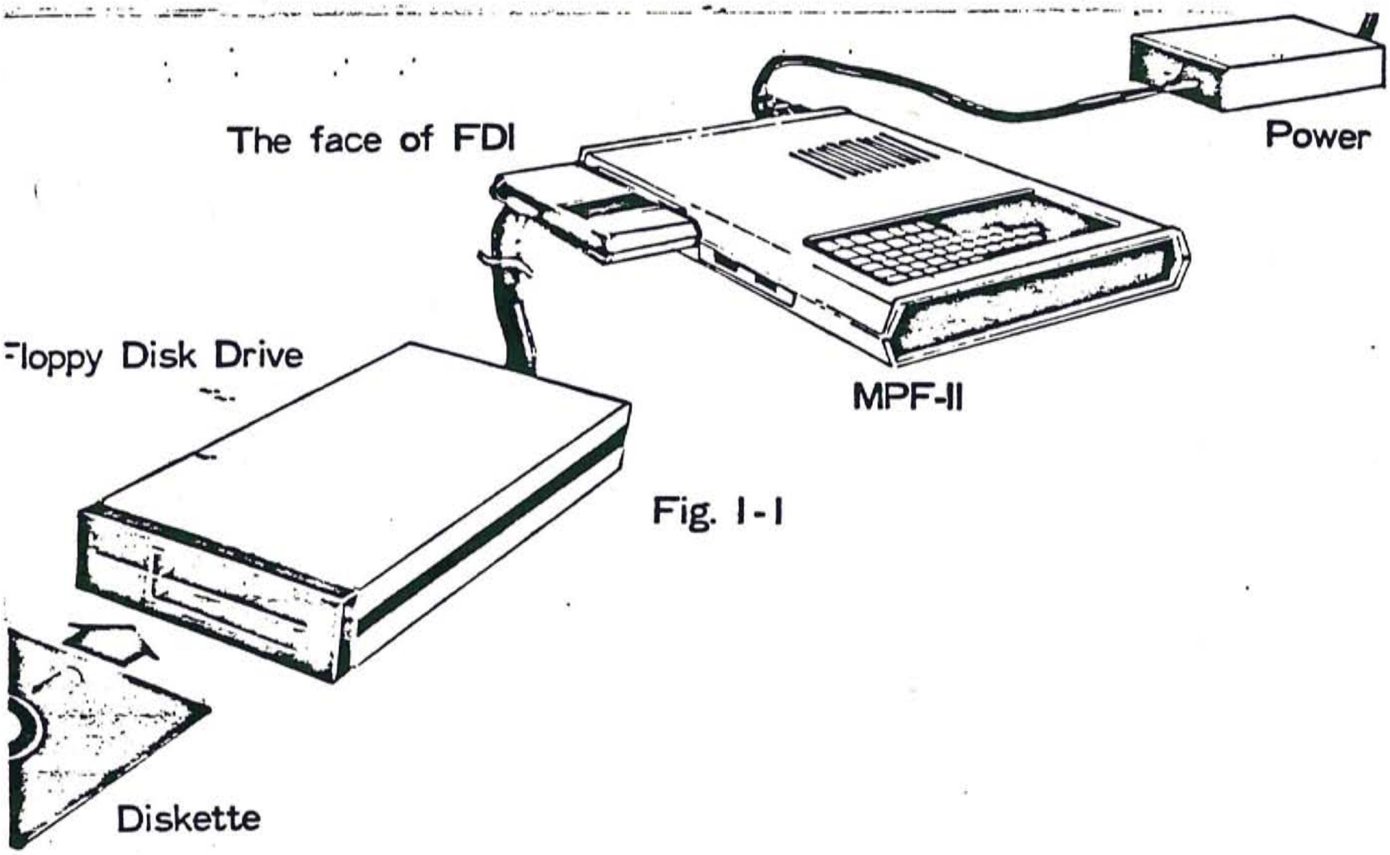

The face of FDI
Power
Floppy Disk Drive
MPF-II
Diskette

Fig. 1-1

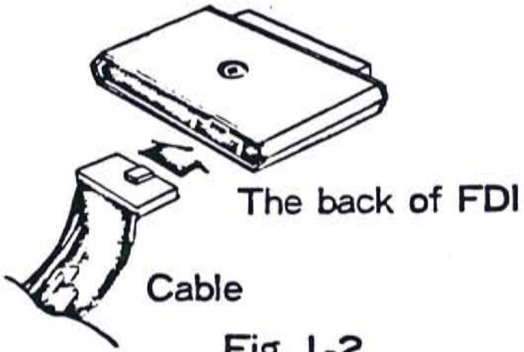

The back of FDI
Cable

Fig. 1-2

2 Convert Your Apple II DOS to MPF-II DOS

1. Set up the disk drive system while power-off.

2. Insert the diskette containing the convertor program into the drive.

3. Turn on the power. The display will show

```
*************************************
* APPLE DOS TO MPF-II DOS CONVERTOR *
*************************************
```

4. When the screen displays

```
"INSERT DISKETTE CONTAINING APPLE DOS
PRESS RETURN KEY TO CONTINUE",
```

   take out the diskette currently in the drive and slide in the diskette containing Apple DOS.

5. Press carriage return key ↵ and the screen will show

```
READING.
```

At this time, the MPF-II is reading in the Apple DOS.

a. If the MPF-II finishes reading correctly, the screen will display

"READ COMPLETE".

You can proceed to step 6.

b. If error message

"APPLE DOS DISKETTE READ ERROR"

shows up on the screen, turn off the power, take out the diskette containing Apple DOS and repeat the reading procedure from step 2.

6. When the screen shows

"PLEASE TAKE ORIGINAL DISKETTE AWAY.
AND INSERT A NEW DISKETTE.
THE NEW INSERTED DISKETTE WILL BE
FORMATTED.
PRESS RETURN TO CONTINUE.",

insert a new diskette.

7. Press the carriage return key ↵ and the screen will display

"FORMATTING".

This is to tell you that the MPF-II is formatting the diskette you just inserted.

a. If the formatting is done successfully, the screen will show

"FORMAT COMPLETE".

You can proceed to step 8.

b. If the screen shows "FORMAT ERROR", repeat from step 6.

8. When the MPF-II starts writing MPF-II DOS, the screen will display

"START WRITING"

a. If the screen shows

"O.K. COMPLETE",

proceed to step 9.

b. If the screen shows

"WRITE ERROR",

repeat from step 6.

9. Turn off the power off and then turn on the power. By doing so, you have finished conversion of Apple DOS to MPF-II DOS.

Note:

1. If the screen gets blurred after you booted the MPF-II DOS with the new, converted master diskette, don't feel bad. Type in the command "HOME" and then press ↵, the blurred display will disappear. Then type in the following HELLO (greeting) program:

```
1Ø TEXT
2Ø PRINT "MPF-II DOS VERSION 2.1 mm/dd/yy"
```

Save this program will the statement SAVE HELLO.

Note the "mm/dd/yy" is the date on which you convert the Apple DOS to MPF-II DOS.

2. After you have made your own MPF-II DOS master diskette, you can use that diskette to boot your DOS the same way as described in Apple II DOS Manual.

3 Convert the COPYA program for MPF-II

1. When prompted by the ">" on the screen, input "BRUN COPYA.COVT" and press the carriage return key ↵.

2. The screen will first get clear and then display

```
          APPLE COPY TO MPF-II COPY CONVERTOR
```

3. When the screen shows

```
          INSERT DISKETTE CONTAINING APPLE COPYA.
          PRESS RETURN KEY TO CONTINUE
```

Put the diskette containing Apple COPYA into the disk drive and press the carriage return key ↵.

After the carriage return key is hit, the MPF-II will be reading the COPYA program. The screen will show

```
          READING
```

4. When the MPF-II finished reading, the screen will display

```
READ COMPLETE
PLEASE TAKE THE ORIGINAL DISKETTE AWAY
AND INSERT A NEW DISKETTE
WHICH HAS BEEN FORMATTED BY
MPF-II DOS OR APPLE DOS 3.3
PRESS RETURN KEY TO CONTINUE
```

In response to the screen message, slide in a diskette which has been formatted by MPF-II DOS or APPLE DOS 3.3 and press the carriage return key ⏎

5. At this moment, the MPF-II is writing the revised COPY program to the diskette just inserted into the disk drive. The screen will show

```
START WRITING
```

6. When the write operation is finished, the screen will show

```
O.K. COMPLETE
```

7. The the converted program is renamed **COPYM.**

8. When you execute the COPYM program itself, you may be prompted by the MPF-II for slot or drive numbers. Enter 1 for both, when prompted, because only one slot and drive are in use.

4 Convert the FID program for MPF-II

1. When prompted by the ">" on the screen, input "BRUN FID.COVT" and press the carriage return key ↵.

2. The screen will first get clear and then display

```
*****************************************
*APPLE FID TO MPF-II FID CONVERTOR*
*****************************************
```

3. When the screen shows

```
INSERT DISKETTE CONTAINING APPLE FID
PRESS RETURN KEY TO CONTINUE
```

Put the diskette containing Apple FID into the disk drive and press the carriage return key ↵.

After the carriage return key is hit, the MPF-II will be reading the FID program. The screen will show

```
READING
```

4. When the MPF-II finished reading, the screen will display

```
READ COMPLETE
PLEASE TAKE THE ORIGINAL DISKETTE AWAY
AND INSERT A NEW DISKETTE
WHICH HAS BEEN FORMATTED BY
MPF-II DOS OR APPLE DOS 3.3
PRESS RETURN KEY TO CONTINUE
```

In response to the screen message, slide in a diskette which has been formatted by MPF-II DOS or APPLE DOS 3.3 and press the carriage return key ⟵

5. At this moment, the MPF-II is writing the revised FID program to the diskette just inserted into the disk drive. The screen will show

```
START WRITING
```

6. When the write operation is finished, the screen will show

```
O.K. COMPLETE
```

7. When you excute the FID program itself, you may be prompted by the MPF-II for slot or drive numbers. Enter 1 for both, when prompted, because only one slot and drive are in use.

Chapter 5 Operating Other Programs

After you boot the DOS, you can try the other programs provided by Multitech.

1. Since a greeting program is already on your MPF-II DOS diskette, each time you boot the DOS the screen should show

   ```
   MPF-II DOS VERSION 2.1 mm/dd/yy
   ```

   On the diskette containing the convertor programs, a HELLO program is also provided. You can run this program by typing

   ```
   >RUN HELLO  <--'
   ```

2. Run the program whose filename is O/X GAME. This program is on the diskette provided by Multitech together with the convertor programs. To try this program, open the disk drive and insert that diskette. Then type

   ```
   >RUN O/X GAME  <--'
   ```

3. COPYM is the program which moves all the files on a diskette to another. This program is on the new diskette which contains the object code converted by the COPY.COVT convertor.

   a. Type

      ```
      >RUN COPYM
      ```

      On the last line of the screen message, you will see

      ```
      PRESS RETURN KEY TO BEGIN COPY
      ```

   b. After typing <--', the last line of the screen will display

      ```
      INSERT ORIGINAL DISK AND PRESS RETURN
      ```

   c. Insert the diskette from which files are to be copied, and press the <--' key. The last line on the screen will show

      ```
      INSERT DUPLICATE DISK AND PRESS RETURN
      ```

   d. Insert a blank diskette and press the <--' key.

   e. Repeat steps b and c until the screen shows

      ```
      DO YOU WANT COPY ANOTHER?
      ```

Press "Y" for "YES", and "N" for "NO".

In the following examples demonstrating how to try the various programs. You can use the CATALOG command to locate the related file (program).

4. FID

Type

```
>BRUN FID  <--'
```

Then the screen will display nine functions. To perform the nine functions, follow the instructions displayed on the screen.

5. NURSE

Type

```
>RUN NURSE <--'
```

Then follow the instructions displayed on the screen to perform a health check for the MPF-II.

6. DEMO-GRAPH

Type

```
>RUN DEMO-GRAPH <--'
```

Then follow the instructions displayed on the screen to examine the graphics capabilities of the MPF-II. You can also use the LIST command to list this program.

7. DEMO-3ØØ-CCC

Note that when you run this program, a CCC should be connected to the MPF-II. Before executing this program, you should follow the procedures listed as follows:

1) Connect the CCC and the FDI to your MPF-II.
2) Boot the DOS while the switch of the CCC is off.
3) Turn on the switch of the CCC.
4) Press the RESET key on the MPF-II keyboard.
5) Enter the monitor by typing CALL-159.
6) Change the values of two memory locations--68 and 1ØØØ--by typing

```
@68:1Ø
@1ØØØ:Ø
@CONTROL C <--'
```

Then you can run the program by typing

```
>RUN DEMO-3ØØ-CCC <--'
```

Then follow the instructions displayed on the screen.

Why change the two values at step 6? The CCC, when used together with MPF-II, will occupy the memory range (of the MPF-II) from memory location 8ØØ (hexadecimal) to FFF. Therefore, the RAM area from 8ØØ to FFF can not be stored with a program. Then, where do we load the program DEMO-3ØØ-CCC? From location 1ØØØ (hexadecimal) and upwards. Because the value at location 68 points to the starting address of the BASIC program to be exeuted, 1Ø is stored at location 68. The program of DEMO-3ØØ-CCC actually starts from location 1ØØ1. The location preceding the starting address of the BASIC program in RAM should always be stored with zero. Therefore, zero is put into location 1ØØØ.

After this program is executed, the program pointer--formed by locations 67 and 68--will contain the value Ø1 and 1Ø, respectively. That means when the monitor searchs for a program to execute, it will search the locations 67 and 68. Please refer to Chapter 6 for a detailed description of memory mapping of the RAM after the MPF-II is connected with the CCC. Thus, you have to follow the procedures described below if you want to execute a program after executing the DEMO-3ØØ-CCC program.

a. Turn off the power of the MPF-II.
b. Turn off the switch on the CCC (Or disconnect the CCC from the MPF-II.)
c. Turn the MPF-II DOS on.
d. Execute the program.

8. DEMO-SONG-SSG

Type

```
>RUN DEMO-SONG-SSG <--'
```

Note that when you run this program, an SSG should be connected to the MPF-II. When this program is being run, music will be generated by the SSG and musical scores will be displayed on the screen.

9. RENUMBER

It is a utility program that can rearrange the statement numbers of a BASIC program. Before running this program, run the program RENUMBER INSTRUCTIONS for a detailed instructions on how to use the RENUMBER utility.

Before running the RENUMBER utility, note that after execution of the RENUMBER utility program, the pointer (which points to the starting location of the program when the monitor program searches a program to execute in RAM) will point to the location 4ØØ1 (hexadecimal).

The locations 67 and 68 are used as program start pointer. The values contained in these two locations are recognized by the monitor-- as the location from where the program to be executed can be fetched. Normally, the values contained in these two locations are 8Ø1 (hexadecimal). Refer to Chapter 6 for a detailed description of the memory map of MPF-II-DOS.

After typing

RUN RENUMBER <--'

Follow the instructions displayed on the screen. Note that after typing in "RUN RENUMBER <--'" the MPF-II will not begin th renumber process until you press the "&" character on the keyboard. Various command tails such as H, M, S, or E can be added to the RENUMBER command "&". (Refer to the instructions printed out by the RENUMBER INSTRUCTIONS on how to add command tails.)

WARNING: You can renumber ONLY BASIC program.
If a program is written in assembly program, wholely or partly, you can not use the RENUMBER utility. Some BASIC program may contains sections of assembly language subroutines. While programming in BASIC, some programmers may use the POKE command to put into a memory block machine code instructions. In this case, the RENUMBER utility may not work.

WARNING: NEVER run a renumbered program immediately after it is renumbered. You should follow the following procedures if you intend to run the renumbered program: 1) Save the renumbered program on diskette. 2) Turn off the power and then turn on the power. 3) Load the renumbered program from diskette into the MPF-II and then run the program. If you don't follow the procedures mentioned above, you are running the risk of ruinning you program.

A second way to run the renumbered program is: 1) Save the renumbered program on diskette. 2) Set the values of two memory locations--68 and 8ØØ--by following the following procedures

Type

```
>CALL-159 (Enter the monitor)
@68:Ø8   (Set the starting address of the BASIC
          program you want to execute to 8ØØ.)
@8ØØ:ØØ (The starting address of the BASIC program
          actually starts from 8Ø1 in RAM. The value
          in the memory location in front of the
          starting address of the BASIC program
          should always be zero.)
```

```
@CONTROL C <--' (Reenter BASIC)
```

3) Execute the renumbered program or other programs.

Note that after using the RENUMBER utility, if you want to execute other programs contained on the diskette provided by Multitech, you must reset the program start pointer--put Ø1 to location 67 and Ø8 to location 68--with the following POKE command:

```
POKE 1Ø4,8
```

(The content of location 67 is 1 constantly.)

1Ø. COPY.COVT

Type

```
>BRUN COPY.COVT <--'
```

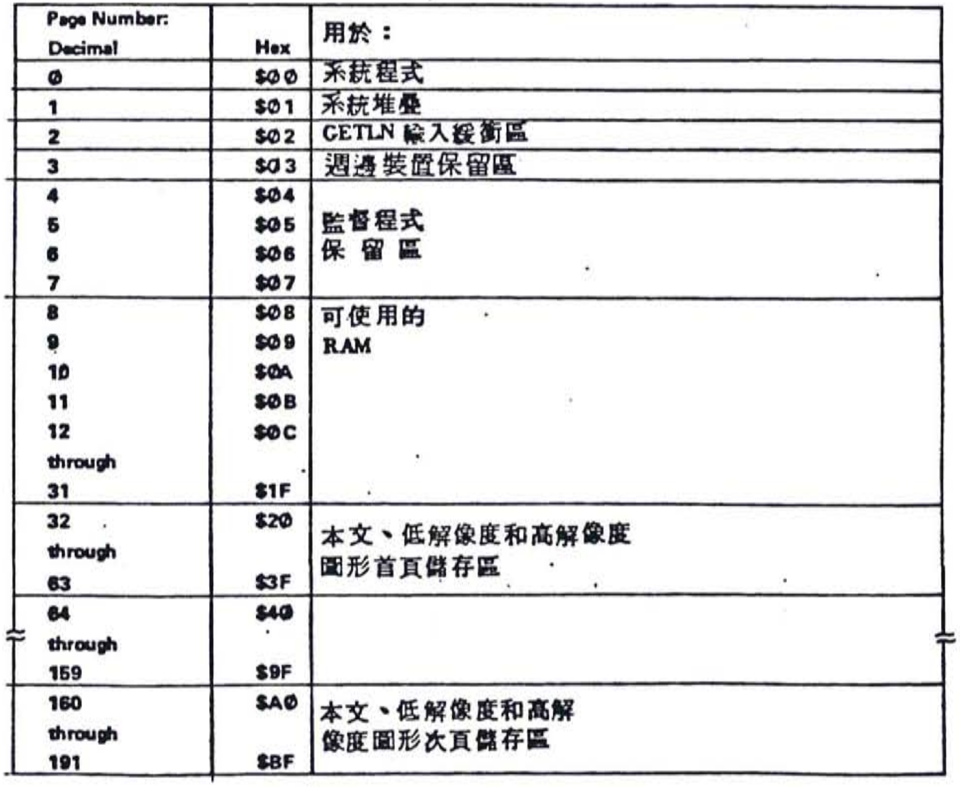
11. FID.COVT

Type

```
>BRUN FID.COVT <--'
```

To enable a user to use MPF-II DOS, a memory map of the MPF-II is provided as. follows (Fig. 6-1):

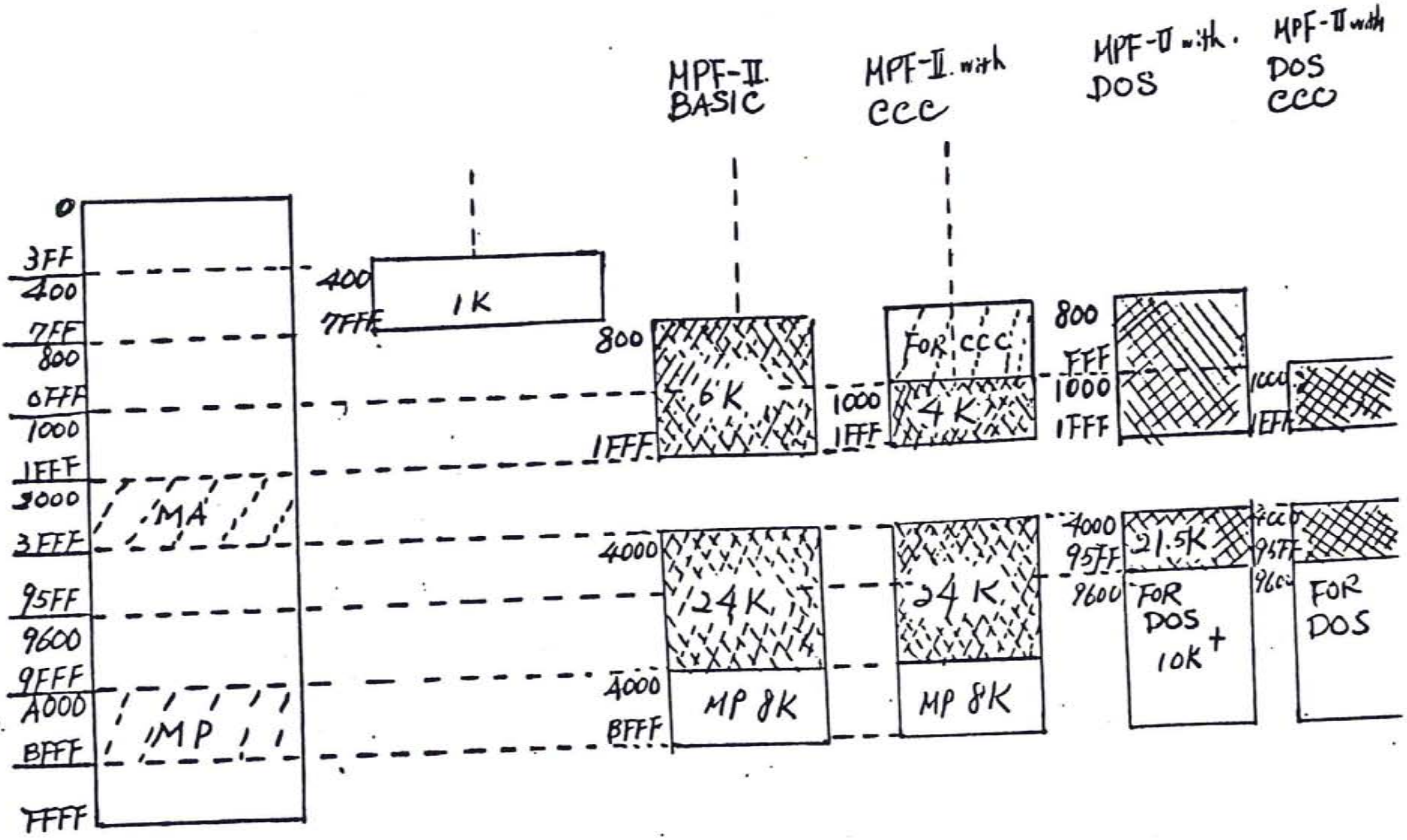
When the MPF-II is connected with a disk drive, or a Chinese Character Controller (CCC), some RAM space will be occupied or used by these peripneral devices. In case your MPF-II is connected to external devices such as a disk drrive, you must understand the exact RAM area which is available to you. Fig. 6-2 provides a comparative memory map of the MPF-II.



Page Number: Decimal	Hex	用於：
Ø	$ØØ	系統程式
1	$Ø1	系統堆疊
2	$Ø2	CETLN 輸入緩衝區
3	$Ø3	週邊裝置保留區
4	$Ø4	監督程式 保 留 區
5	$Ø5	
6	$Ø6	
7	$Ø7	
8	$Ø8	可使用的 RAM
9	$Ø9	
1Ø	$ØA	
11	$ØB	
12	$ØC	
through		
31	$1F	
32	$2Ø	本文、低解像度和高解像度 圖形首頁儲存區
through		
63	$3F	
64	$4Ø	
through		
159	$9F	
160	$AØ	本文、低解像度和高解 像度圖形次頁儲存區
through		
191	$BF	

Fig. 6-1

~~表十~~、RAM組織和用途



Shaded Area is available RAM area to users

Fig. 6-2

6.1 Ram Available to a User --When the MPF-II stands alone

When the MPF-II is not connected with the CCC or disk drive, the RAM area available to a user (the area which you can write your program) is the memory range from 800 (hexadecimal) to 1FFF and the memory range from 4000 to BFFF. From Fig. 6-2, you will see that the RAM area available is separated.

Note that the area from 2000 to 3FFF is used as the primary screen display buffer (or page 1), and the RAM area from A000 to BFFF is also used as the secondary display buffer (or page 2). If the secondary page is not used, then data and program can be stored in this area.

If your program is less than 6K bytes, you can store your program in the memory block ranging from 800 to 1FFF. If your program takes more than 6K bytes of space, you can store the program after typing in the MP command (A special MBASIC command).

The MP Command

What the MP command does is to set the primary display buffer to the area from A000 to BFFF and make the area from 2000 to 3FFF available as a RAM area where you can store your program. After typing in the MP command, you can store your program beginning from the memory location 800 to 9FFF, totaling 38K bytes.

Another way to store program of longer than 6K bytes is to store the program into the RAM area beginning from 4000 to 9FFF, totaling 24K bytes. Using this method, a programmer can only store the program which takes less than 24K bytes of space.

As soon as the power to the MPF-II is turned on, the values of two memory locations--67 (hexadecimal) and 68--are set to 01 and 08, respectively. The contents (values) of the two locations serve as a pointer which points to the location (address) from where the program to be executed can be fetched. The byte that precedes the starting address of the program to be executed should always be stored with the value "0". Thus, a BASIC program to be executed normally is stored in the RAM beginning from location 801 (hexadecimal). The byte preceding 801--800--should always be stored with zero.

Using the POKE Command

To store a BASIC program starting from 4001 in RAM, you have to use the following commands

POKE 103,01 (Put 1 to location 67--The decimal equivalent to 67 is 103.)

POKE 104,64 (Put 40 to location 68--The decimal equivalents to 40 and 68 are 104 and 64 respectively.)

POKE 16384,0 (Put 0 to location 4000--The decimal equivalent to 4000 is 16384.)

For details of the POKE command, refer to the MPF-II BASIC Programming Manual.

6.2 RAM Available to a User --When the MPF-II is connected with the Chinese Character Controller (CCC)

When the MPF-II is connected with the Chinese Character Controller (CCC), the RAM available to a user is the two memory blocks--one from 1000 to 1FFF and the other from 4000 to BFFF. Since the two memory blocks are separated by the primary display buffer, and the RAM area from 1000 to 1FFF can only store 4K bytes, you have to use the MP command or the POKE command as mentioned above when your program needs more than 4K bytes of space.

6.3 RAM Available to a User --When the MPF-II is connected with the Disk Drive

WARNING: When your MPF-II is connected with a disk drive, it will not respond to the MP command.

When your MPF-II is connected with a disk drive, the RAM area from 9600 to BFFF will be occupied or used by the MPF-II DOS. The RAM area available to a programmer are two separate memory blocks--one from 800 to 1FFF and the other from 4000 to 95FF. Remember you can use either the RAM area from 800 to 1FFF, which totals 6K bytes, or the area from 4000 to 95FF, which totals 21.5K bytes.

Considering the facts that MP command is invalid and that the RAM area from 800 to 1FFF can only store 6K bytes, the maximum RAM area available to a user is 21.5K bytes. If you write a program which takes more than 21.5K bytes, you may lose your program.

6.4 RAM Available to a User --When the MPF-II is connected with the Disk Drive and the CCC

When the MPF-II is connected with the Disk Drive and the CCC, the RAM area available to a user is two separate memory blocks--one from 1ØØØ to 1FFF (4K bytes) and the other from memory location 4ØØØ to 95FF (21.5K bytes).

When you write a program, make sure it does not exceed the 21.5K-byte limit. Use the POKE command to store your program into the RAM beginning from 4ØØØ.

在此將DOS命令歸納爲五大類：

Appendix: MPF-II DOS Command Summary

管理類命令 Housekeeping Commands

INIT
CATALOG
SAVE
LOAD
RUN
RENAME
DELETE
LOCK
UNLOCK
VERIFY
MON
NOMON
MAXFILES

取存類命令 Access Commands

FP
CHAIN

順序文字資料檔案命令 Sequential Text File Commands.

OPEN
CLOSE
READ
WRITE
APPEND
POSITION
EXEC

隨機取存文字資料檔案命令 Random Access Text File Commands

OPEN
CLOSE
READ
WRITE

機器語言檔案命令 Machine Language File Commands

BSAVE
BLOAD
BRUN

GENERAL GUIDE LINE TO CONVERT THE APPLE-II MACHINE PROGRAM
TO MPF-II PROGRAM:

1.Find the instruction "LDA 0C000H" in the APPLE-II machine program and replace it by the following instructions in the MPF-II machine program:

```
LDA $26
PHA
LDA $27
PHA
LDA $06
PHA
LDA $07
PHA
LDA $08
PHA
LDA $09
PHA
JSR $F043
PHA
TXA
PHA
TSX
TXA
EOR #$80
TAX
PLA
STA $0100,X
INX
PLA
STA $0100,X
PLA
STA $09
PLA
STA $08
PLA
STA $07
PLA
STA $06
PLA
STA $27
PLA
STA $26
LDA $0100,X
PHA
DEX
LDA $0100,X
TAX
PLA
RTS
```

2.Secondary Screen display:

Find all the instructions which refer to the secondary screen buffer of APPLE-II (0400H to 05FFFH) and change them to the corresponding address of the MPF-II secondary screen buffer (0A000H to 0B000H).

3.Text mode display: find the instructions used to display the APPLE-II text: "LDA @0C1H STA 0401H" and convert them to the following instructions in the MPF-II machine profram:

```
LDA @001H ;X axis
STA 024H
LDA @000H ;Y axis
STA 025H
LDA @0C1H
JSR 0FDF0H
```

MULTITECH ELECTRONICS INC.

195 W. EL CAMINO REAL • SUNNYVALE, CA 94086 • (408) 773-8400

MPF-II MEMORY MAP

Memory Range	Description
0.—1 FF	Reserve for system,Single Monitor.
200.—2FF	Keyboard buffer.
300.—3FF	Machine code Program.
400.—4FF	System Variables,special Ultilities Program.
800.—1FFF	MBASIC Program and Variables.
2000.-3FFF	Whem Power ON or MA command,these memory are reserved for Low/High resolution and text mode.
4000.-9FFF	application program,Data and variables area.
A000.-BFFF	MP Command.these memory are reserved for low/high resolution and text mode.
C000.-FFFF	I/O port,system monitor program area.

THE MEMORY LOCATION OF RAM OCCUPIED BY THE SCREEN

	APPLE-II	MPF-II
TEXT PRIMARY BUFFER	400H-7FFH	2000H-3FFFH
Text Secondary Buffer	800H-8FFH	A000H-BFFFH
Low Resolution Primary	400H-7FFH	2000H-3FFFH
Low Resolution Secondary	800H-8FFH	A000H-BFFFH
High-resolution primary	2000H-3FFFH	2000H-3FFFH
High-Resolution secondary	4000H-5FFFH	A000H-BFFFH

The Difference in BASIC Commands

	APPLE-II	MPF-II
FLASH	YES	NO
IN#	YES	NO
PR#	YES	NO
MA	NO	YES
MP	NO	YES
PRTON	NO	YES
PRTOFF	NO	YES
HC	NO	YES
GR,HGR	Clear the screen	Clear the window only.

MULTITECH ELECTRONICS INC.

195 W. EL CAMINO REAL • SUNNYVALE, CA 94086 • (408) 773-8400

LOADING APPLE-II TAPES INTO MPF-II

Set up : Use any Cassette Recorder, volume and tone control are set at mid-high range.

Procedure : After power on, type MP command, then CR screen shows the bar pattern similar to that immediately after turn on power. This pattern stays through the LOADA process. It clears after RUN & CR command. Also, the cursor is half height.

APPLE-SOFT PROGRAM	LOADA	RUN
1. Penny Arcade	O.K.	Text display O.K. Partial picture Bouncing ball sound O.K.
2. Finance I	O.K.	O.K.
3. Little Brick out	O.K.	Text O.K. (No paddle to use)
4. Color Demosoft	O.K.	Choice 1 & 2 picture flashes then returns to Menu choice 3 & 4. (No picture)
5. Hopalong cassity	O.K.	O.K.
6. Lemonade	O.K.	Syntax error in 10010
7. Phone list	O.K.	O.K.
8. Brians Theme	O.K.	O.K.
9. Alignment test tone	Errerr	Can not load into APPLE either
10. Renumber/Append	O.K.	Starts O.K. then entered monitor mode.

MULTITECH ELECTRONICS INC.

195 W. EL CAMINO REAL • SUNNYVALE, CA 94086 • (408) 773-8400

MPF_II MEMORY MAP

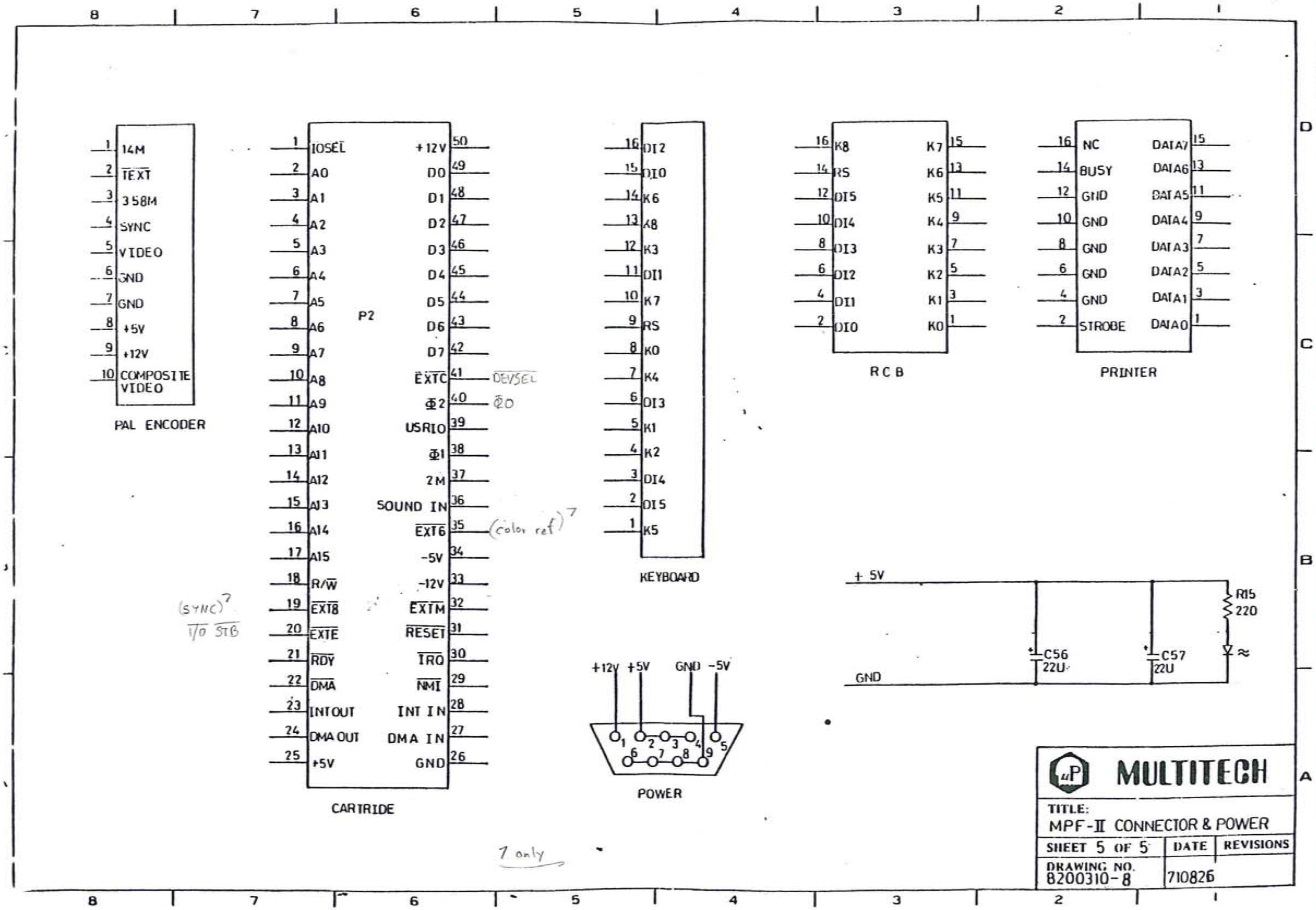
Memory Range	Description
0. 1 FF	Reserve for system,Single Monitor.
200. 2FF	Keyboard buffer.
300. 3FF	Machine code Program.
400. 4FF	System Variables,special Ultilities Program.
800. 1FFF	MBASIC Program and Variables.
2000.3FFF	Whem Power ON or MA command,these memory are reserved for Low/High resolution and text mode
4000.9FFF	application program,Data and variables area.
A000.BFFF	MP Command.these memory are reserved for low/high resolution and text mode.
C000.FFFF	I/O port,system monitor program area.

THE MEMORY LOCATION OF RAM OCCUPIED BY THE SCREEN

	APPLE-II	MPF-II
TEXT PRIMARY BUFFER	400H-7FFH	2000H-3FFFH
Text Secondary Buffer	800H-8FFH	A000H-BFFFH
Low Resolution Primary	400H-7FFH	2000H-3FFFH
Low Resolution Secondary	800H-8FFH	A000H-BFFFH
High-resolution primary	2000H-3FFFH	2000H-3FFFH
High-Resolution secondary	4000H-5FFFH	A000H-BFFFH

The Difference in BASIC Commands

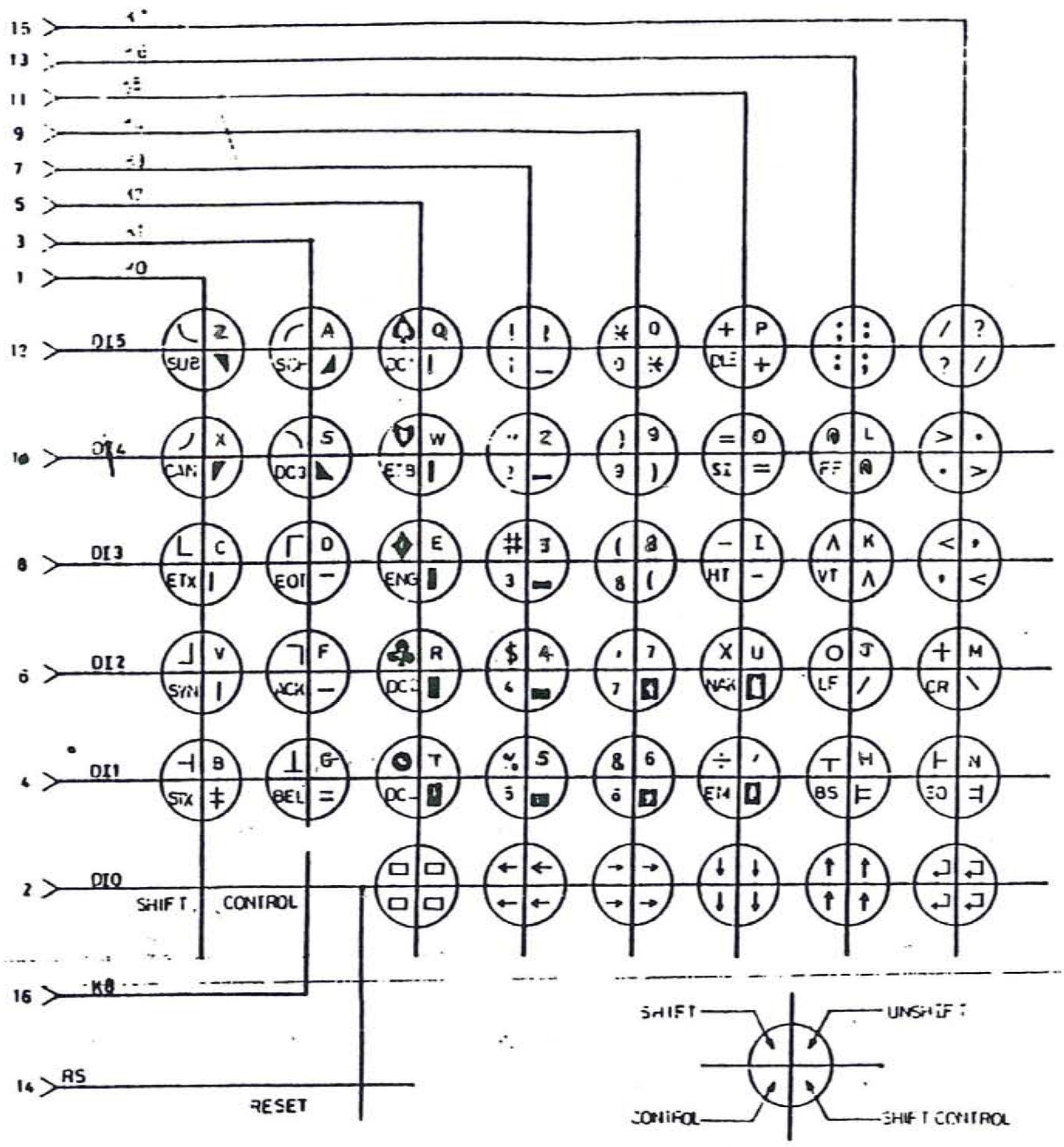
	APPLE-II	MPF-II
FLASH	YES	NO
IN#	YES	NO
PR#	YES	NO
MA	NO	YES
MP	NO	YES
PRTON	NO	YES
PRTOFF	NO	YES
HC	NO	YES
GR,HGR	Clear the screen	Clear the window only.



MULTITECH

TITLE: MPF-II CONNECTOR & POWER

SHEET 5 OF 5	DATE	REVISIONS
DRAWING NO. 8200310-8	710826	



(Confidential)

MULTITECH

TITLE MPF-I KEYBOARD (EXTERNAL)

SHEET 1 OF 1 | DATE | REVISIONS

DRAWING NO. 8200360-1 | 711014

MULTITECH ELECTRONICS INC.

195 W. EL CAMINO REAL • SUNNYVALE, CA 94086 • (408) 773-8400

NOTE FOR ASSEMBLY LANGUAGE PROGRAMMING

1.Monitor subroutine entry:

All the monitor subroutine entry on the Chapter 3 of APPLE-II manual is the same as MPF-II except "KEYIN" and "PREAD" which MPF-II does not support. and the Address of "SCRN" is $F871 in Apple-II and is $F869 in MPF-II.

2.Zero Page Usage:

MPF-II id the same with APPLE-II except the four memory location- '6','7','8','9' are used as TEMP buffer.

"21H" represents :window width in the APPLE-II. for MPF-II,it represents :the window right margin.

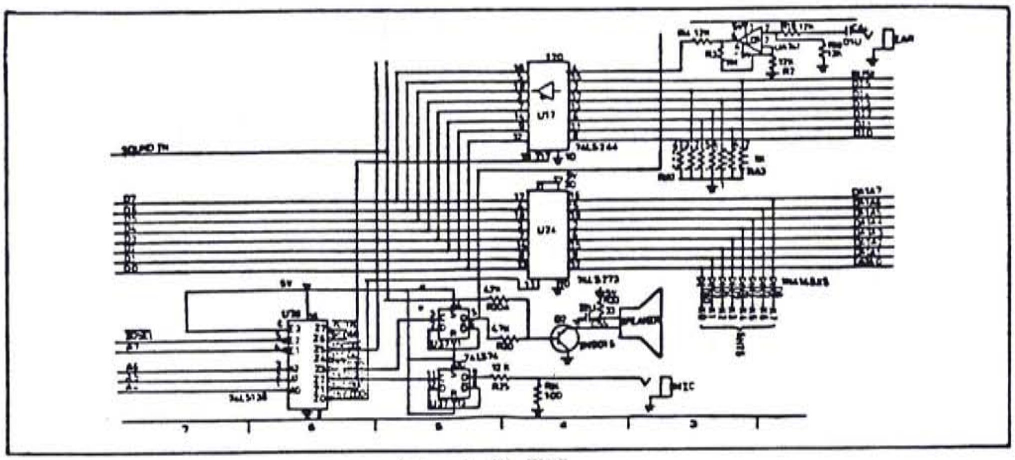
3,Text area:

Some parts of text area are used as system parameter in the MPF-II. MPF-II can't display the APPLE-II assembly program which is written directly into the text area.

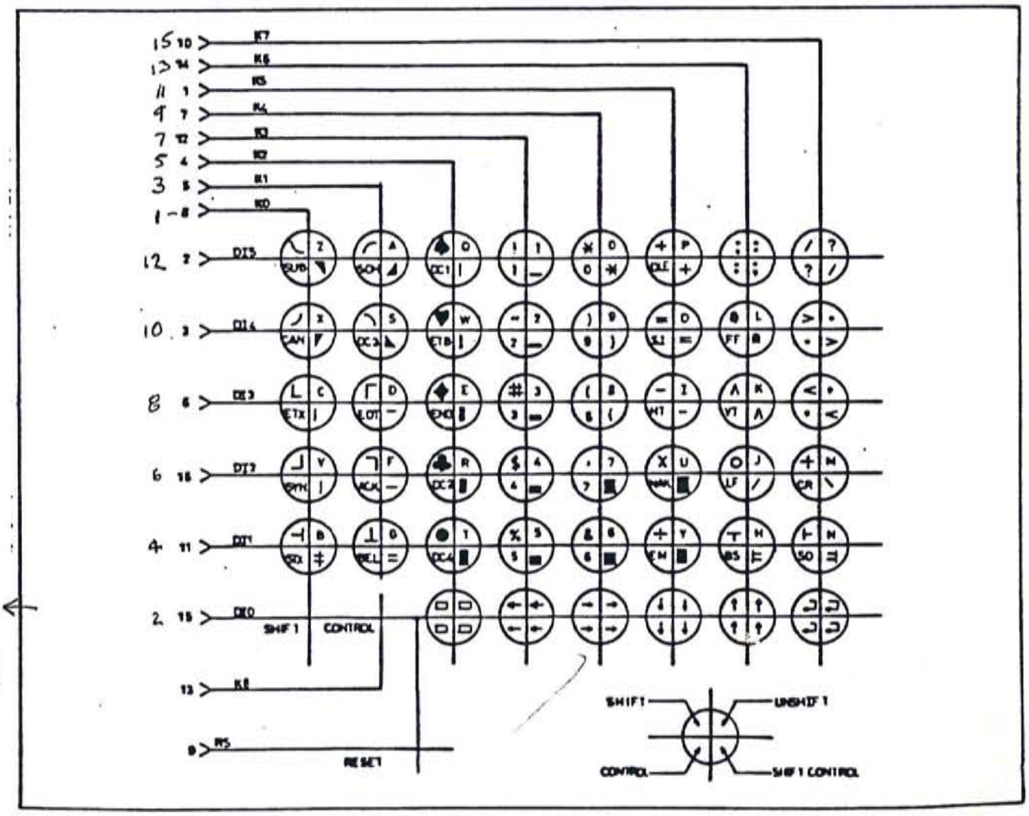
4.Key Reading:

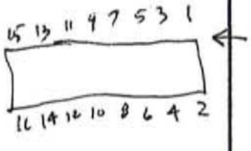
In the MPF-II,using the command JSR F043H (JSR SCAN1) instead of LDA 000H in the APPLE-II for scanning keyboard.

SCHEMATIC



Keyboard





MPF-II Keyboard

Your guide to the world of microprocessors.

The Micro-Professor™ -A low cost tool for learning, teaching and prototyping.

Here in one attractive package and at a price of only $149.00 is a Z80* based microcomputer to lead you step by step to a thorough knowledge of the world of microprocessors.

The Micro-Professor is a complete hardware and software system whose extensive teaching manual gives you detailed schematics and examples of program code. A superb learning tool for students, hobbyists and microprocessor enthusiasts, as well as an excellent teaching aid for instructors of electrical engineering and computer science courses.

But the Micro-Professor is much more than a teaching device. With it you can do breadboarding and prototyping, designing your own custom hardware and software applications with Z80, 8080 and 8085 compatible code.

The standard 2K bytes of RAM is expandable to 4K, and the standard 2K bytes of ROM can be increased to 8K.

All this plus a built-in speaker, a cassette interface, and sockets to accept optional CTC/PIO. Bus is extendable.

As well as being an exciting learning tool, the Micro-Professor is a great low-cost board for OEM's. Call for details.

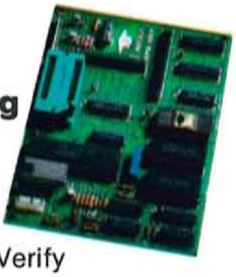
* Z80 is a trademark of Zilog Inc.

MICRO-PROFESSOR
The Bipolar Digital Integrated Circuits Data Book
TTL Bipolar Memory & Interface Circuits
Zilog 1981 Data Book
Zilog
MICRO-PROFESSOR
USER'S AND EXPERIMENT MANUAL
μP
AN EYE TO THE FUTURE
uPF--1

Multitech Electronics Inc.

SSB-MPF Speech Synthesizer Board $129

A vocabulary of up to 400 words based on the TMS 5200 chip.



EPB-MPF EPROM Programming Board $169

For all +5V 1KB/2KB/ 4KB EPROMs. Read/Copy/List/Verify Capability.

BASIC-MPF Tiny Basic $19

2KB BASIC interpreter with hardware control capability. Machine-code subroutine accessible.

I'm ready to enter the world of microprocessing

Call toll free to order.

	Price	Qty.	Amount
MPF-I Micro-Professor	$149.00		
SSB-MPF Speech Synthesizer Board	$129.00		
EPB-MPF EPROM Programming Board	$169.00		
BASIC-MPF	$ 19.00		
Shipping and Handling	$ 4.95		4.95
California residents add sales tax.			
TOTAL			

□ Check or money order enclosed

□ Visa □ Mastercharge

Card No. _______ Expires

Signature _______

Name (Please Print) _______

Address _______

City _______ State _______ Zip

In U.S. and Canada mail to:

Multitech Electronics Inc.
195 West El Camino Real
Sunnyvale, CA 94086

California Tel. (408) 773-8400
Elsewhere (800) 538-1542

Outside of North America mail to:

Multitech Industrial Corporation
977 Min Shen E Road, 105
Taipei, Taiwan, ROC

Tel. 02-769-1225
TWX 19162 MULTIIC,
23756 MULTIIC

Features and Specifications

MPF-1

CPU Z80 CPU high performance microprocessor with 158 instructions.

RAM 2K bytes expandable to 4K bytes.

ROM 2K bytes of sophisticated monitor expandable to 8K bytes.

Input/Output 24 system I/O lines.

Monitor 2K bytes of sophisticated monitor. Monitor includes system initialization, keyboard scan, display scan, tape write and tape read.

Display 6-digit, 0.5″ red LED display.

Audio Cassette Interface 165-bit/sec average rate for data transfer between memory and cassette tape.

Extension Connectors All buses of CPU, channel signals of CTC, and I/O port bus of PIO usable for expansion.

Counter Timer Circuit Socket is provided.

Parallel I/O Circuit Socket is provided.

Speaker & Speaker Driver Circuits 2.5″ diameter speaker

User Area Provides a 3.5″ × 1.36″ wire wrapping area for user's expansion.

Power Requirement 9V, 0.5A adaptor is provided.

User's Manual Complete self-learning text with experiments and applications.

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

RS Reset the system.

ADDR Set memory address and display content.

DATA Input data to memory or register.

PC Recall program counter.

REG Select register and display contents of register.

+ Display content of next memory or register.

– Display content of last memory address or register.

STEP Single step execution of user's program.

SBR Set break point of user's program.

MONI User's program break and return to monitor.

GO Execute the user's program.

INS Insert data of the address followed by the current display address.

DEL Delete data of the current display address.

MOVE Move memory block in the RAM.

RELA Relative address calculation. Calculates and stores relative address.

TAPE WR Store data to the cassette tape.

TAPE RD Load data from the recorder.

INTR Maskable interrupt.

USER KEY User defined key.

O-F Hex-digits or register selection.

EPB-MPF Specifications

Hardware Specifications

Compatible with MPF-1. Use 40-pin flat ribbon cable and male connector to interface with MPF-1.

ROM: Single +5V EPROM 2516 × 1. Total of 2K bytes. Monitor EPROM address: 9000—97FF.

RAM: Static RAM, 6116 × 2. Total of 4K bytes. Basic RAM address: 8000—8FFF.

I/O Port: Programmable I/O port, 3255 × 1. Total of 24 parallel I/O lines. I/O address: CC-CF

System Power Consumption: 25V/30mA and 5V/350mA

Main Power Input: 30V/75mA and 9V/400mA adaptor is provided. Power adaptor input 110V.

Textool: 24-pin, zero insertion force socket.

Software Specifications

READ: Read data from EPROM onto RAM buffer.

VERIFY: Verify EPROM data with RAM buffer.

LIST: Display or modify data on RAM buffer.

RESTART: Restart to initial state of EPB-MPF.

PROGM: Write data from RAM buffer to EPROM.

DEL: Delete data from the current display address in RAM buffer.

INS: Insert data onto the address followed by the current display address of RAM BUFFER.

SSB-MPF Features

- Uses high reliability TI speech synthesis chip.
- 4KB EPROM for time-clock program and speech utility.
- Two EPROM sockets for expanding speech vocabulary.
- Shares the Z80 CPU of MPF-1 as host controller.
- Uses keyboard and speaker of MPF-1 as input/output device.
- Adjustable voice pitch and volume.
- 9V, 0.5A adaptor is provided.
- Complete accessories including 40-pin, double-headed connector, audio jumper, operation manual, etc.

Multitech Electronics Inc.
195 W. El Camino Real, Sunnyvale, CA 94086 (408) 773-8400