

GoldStar

FAMILY COMPUTER

OPERATING MANUAL MODEL FC-200



*MSX is the Trade Mark of Microsoft Co.

*It must be used for the Personal Computer attached the Trade Mark of MSX.



GoldStar

copyright
© 1984
GoldStar Co., Ltd.

NOTICE

The information contained in this manual is for reference only and subject to change without notice.

WARNING: This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instruction manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

WARNING : TO PREVENT FIRE OR SHOCK HAZARD, DO NOT EXPOSE THIS APPLIANCE TO RAIN OR MOISTURE.

CAUTION : TO PREVENT ELECTRIC SHOCK DO NOT USE THIS (POLARIZED) PLUG WITH AN EXTENSION CORD, RECEPTACLE OR OTHER OUTLET UNLESS THE BLADES CAN BE FULLY INSERTED TO PREVENT BLADE EXPOSSURE.

PREFACE

Thank you for purchasing a GOLDSTAR Personal Computer. This computer is designed to use MSX BASIC Language giving access to a wide range of educational games and business software. It is suggested that you read this operating manual to familiarise yourself with the features available.

CONTENTS

INTRODUCTION	
1. PRECAUTIONS	3
2. FC-200 SYSTEM CONFIGURATIONS	5
3. THE NAMES OF THE FC-200S COMPONENTS	6
4. CHECK LIST OF ITEMS	7
5. SETTING UP THE COMPUTER	7
6. OPERATING THE FC-200	8
7. GETTING STARTED	9
8. THE ORDER OF OPERATING THE FC-200	18
9. THE KEY NAMES AND OPERATING METHOD OF THE KEYBOARD ..	19
10. HOW TO USE THE EXCLUSIVE DATA RECORDER AND HOME CASSETTE TAPE RECORDER	28
11. HOW TO ADJUST THE MONITOR AND THE COLOR T.V.	34
12. HOW TO MAKE THE PROGRAM	34
13. HOW TO DRAW THE ANIMATION	71
14. MAINTENANCE AND AFTER-SERVICES	83
15. EXPLANATION OF COMMANDS	85
16. APPENDIX	94

INTRODUCTION

This manual is designed to help anyone unfamiliar with computers to set up the GoldStar and to give examples of the features available. Full details of the very powerful MSX BASIC Language is given in a separate manual which accompanies the computer.

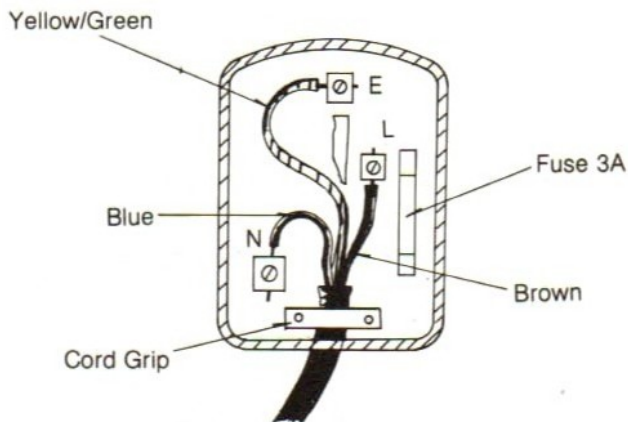
1. PRECAUTIONS

Your GOLDSTAR computer is a robust piece of equipment employing solid state electronic components, but a few precautions should be observed to avoid accidents damage or injury to the user.

1.1 WIRING THE MAINS PLUG (This wiring is applied in BS Area)

Great care should be taken to connect the mains plug correctly. There are three wires in the mains lead, one coloured brown, one blue and one yellow and green. Cut the wire lengths as shown in sketch and connect the brown wire to the plug terminal marked L for 'Live', the blue wire to the terminal marked N for 'Neutral' and the yellow and green wire to the terminal marked E or '⏏'

If there is a fuse in the plug, make sure that this is rated at '3A' (3 amp). The outer sheath of the wire must be secured by the cord grip. Before screwing the top back on to the plug, make sure that the coloured wires are arranged so that they are not trapped by the plug cover.



1.2 GENERAL CARE

Avoid dropping the computer or subjecting it to knocks. Avoid leaving heat. When cleaning use only a cloth very slightly moistened with water (or water and a mild detergent). DO NOT use any form of spirit or solvent.

1.3 ELECTRICAL CARE

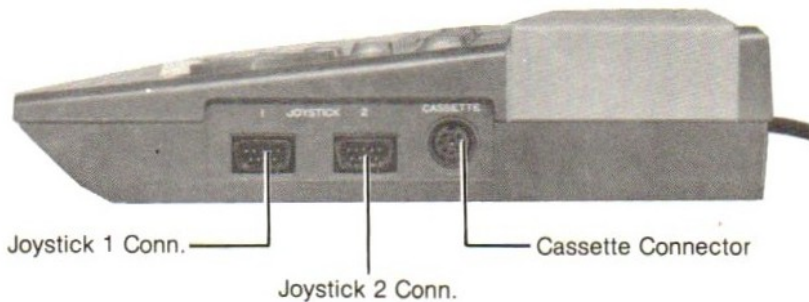
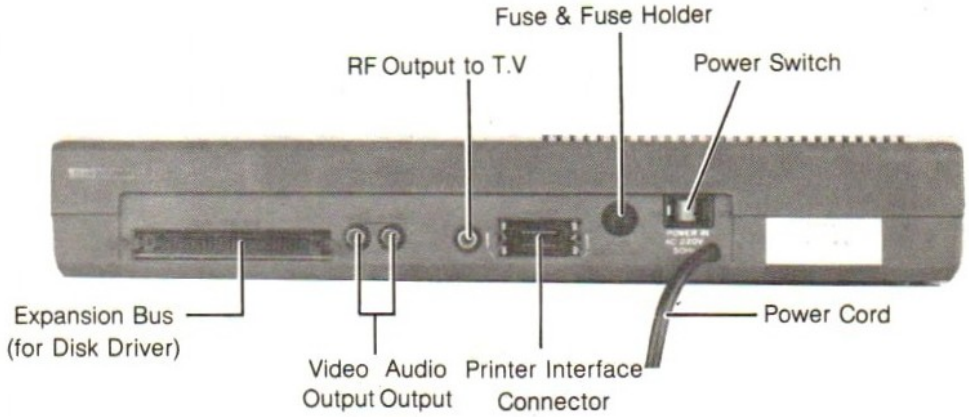
Do not connect or disconnect any cartridge or other plug-in device whilst the computer is switched on. Doing so can damage both the computer and the device being connected or disconnected. If for some reason you wish to switch the computer OFF and ON again, wait at least five seconds between switching OFF and ON. Always remove the mains plug from the wall socket when the computer is not in use.

2. THE FC-200 SYSTEM CONFIGURATIONS



* Color T.V, Color Plotter Printer (PRT-5), Data Recorder (FCD-10B), Joystick, Game Pack and Adaptor for Data Recorder and sold respectively.

3. THE NAME OF THE FC-200'S COMPONENTS



4. CHECK LIST OF ITEMS

You should check that the following items are in the package.

1. Computer (with power cord)
2. Operating Manual and Basic Manual
3. Connection cord for Cassette Tape Recorder
4. Monitor or TV connection cord

5. SETTING UP THE COMPUTER

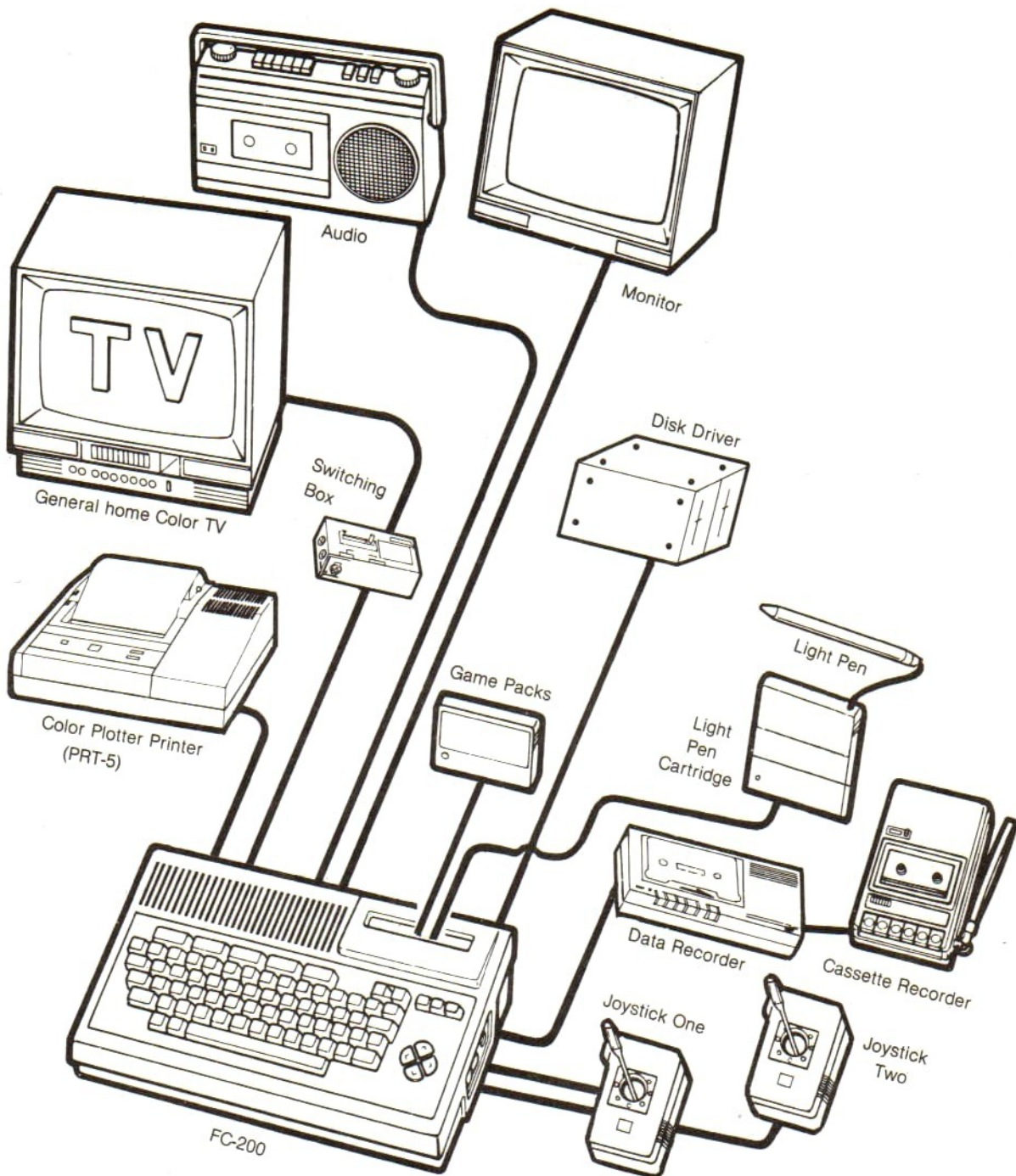
The following diagram shows how the various components available for the GOLDSTAR connect together.

The power must be OFF when any of the components (peripherals) are connected or disconnected. Connect a lead between the output socket marked 'RF' and the aerial socket of the T.V.

After the computer has been switched ON it will be necessary to tune the chosen T.V. channel to around channel 36 and then fine tune until a good screen display is obtained.

6. OPERATING THE FC-200

The following shows the overall system configuration of the FC-200.
(Turn off the power of all the peripherals and the host, in case of connecting, and refer to the articles how to connect the host with the peripheral devices).



7. GETTING STARTED

After connecting a T.V. and tuning to channel 36, switch on the computer.

7-1.1 THE INITIAL MESSAGE ON THE SCREEN

An initial message like figure A will appear on the screen and shortly afterwards will be replaced by the message shown at B. (If the TV was not tuned correctly, message B will already be on the screen by the time the TV has been fine-tuned).

The appearance of message B shows that the computer has powered up correctly.

NOTE: If a ROM cartridge or Game Pack has been inserted, a different message or display will appear.

After display B appears on the screen, the computer is ready to accept a program.

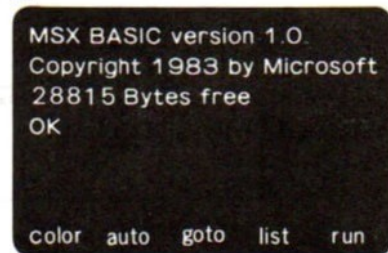
THIS CAN BE DONE EITHER:

1. Entering the program by keyboard
2. By loading from a cassette tape

A



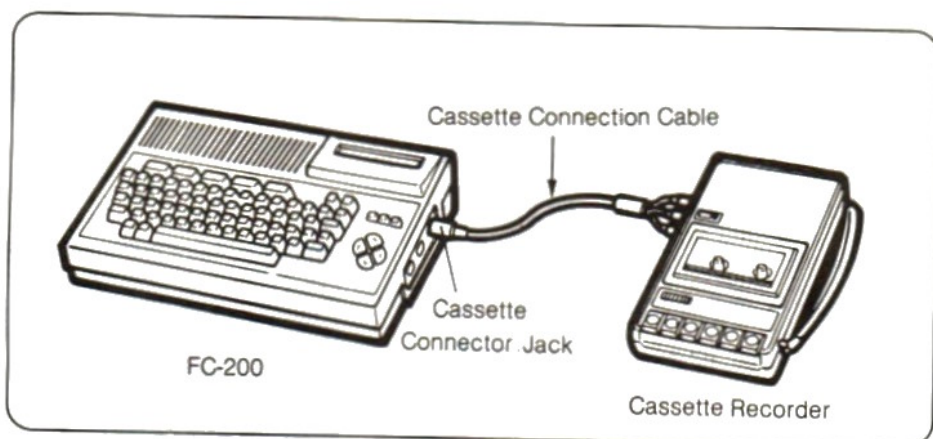
B



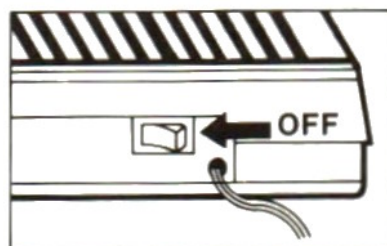
7-2. CONNECTING TO THE SYSTEM THE GENERAL HOME CASSETTE RECORDER.

You may connect the Cassette Recorder to the FC-200 with the Cassette Recorder connection cable.

If there is no remote (REM) terminal on Cassette Recorder or the size of the remote does not suit, you don't have to connect the plug for the remote terminal (grey). But, in this case, the function of AUTO START/STOP won't be performed.



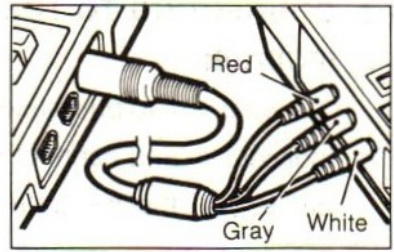
1. Turning the Power Switch to the OFF Position
Reach behind the FC-200 and turn the ON/OFF switch to the OFF position. Connecting with the power switch ON might cause miss operation and breakdown.



2. Connecting the Cassette Recorder Input Terminal to the System

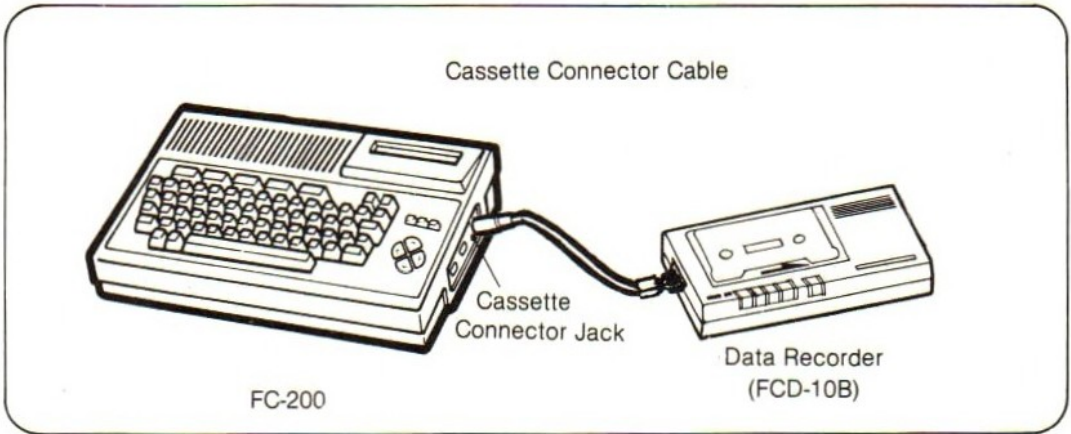
Connect the input terminal of the cassette recorder to the cassette terminal on the side of the host with the cassette connection cable.

- o Connect the white terminal to the "EAR" terminal of the cassette recorder.
- o Connect the red terminal to the "MIC" terminal of the cassette recorder.
- o Connect the gray terminal to the "REM" of the cassette recorder.

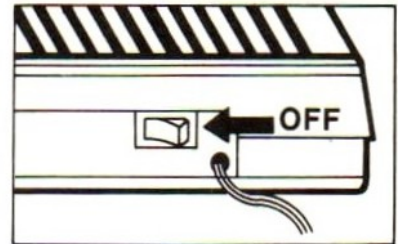


7-3. CONNECTING TO THE SYSTEM THE EXCLUSIVE DATA RECORDER (FCD-10B)

We want you to purchase the exclusive data recorder for FC-200 (FCD-10B) individually and connect it to the system with the data recorder connection cable. (i.e. the cassette connection cable) If you use the exclusive data recorder, the automatic operation of a tape is possible.



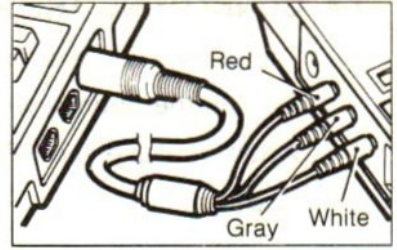
1. Turning the Power Switch to the OFF Position
Reach behind the system and turn the ON/OFF switch to the OFF Position. Connecting with the power switch ON might cause miss operation and breakdown.



2. Connecting the Exclusive Data Recorder

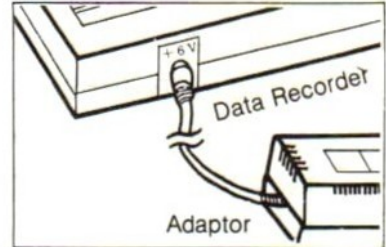
Connect the input terminal of the data recorder to the cassette terminal on the side of the host with the cassette connection cable.

- o Connect the white terminal of the data recorder to the "CMTOUT" terminal (WHITE).
- o Connect the grey terminal of the data recorder to the "REM" terminal (GREY).
- o Connect the red terminal of the data recorder to the "CMT IN" terminal (RED).



3. Connecting DC +6V Adaptor

We want you to purchase the separate adaptor (GA-60) and connect it to the DC +6V terminal on the back panel of the data recorder.

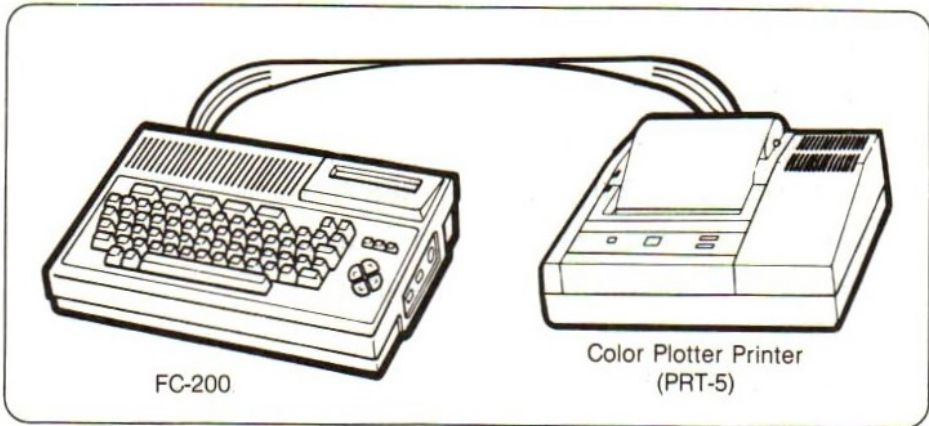


7-4. CONNECTING THE PRINTER (PRT-5) TO THE SYSTEM

The printer for Centronics standard is used.

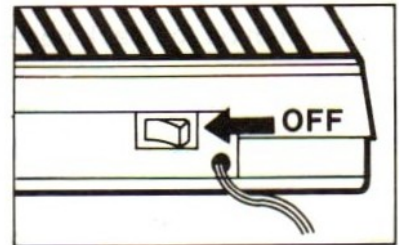
Consult the agency where you purchased about the type of the printer.

When you use the PRT-5 (Gold Star Color Plottor Printer), Connect it as follows:



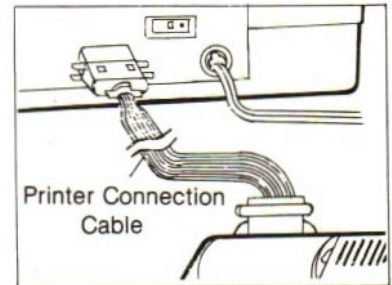
1. Turning the Power Switch to the OFF Position

Reach behind the system and turn the ON/OFF switch to the OFF Position. Connecting with the power switch ON might cause miss operation and breakdown.



2. Connecting the System to the PRT-5

Connect the system to the with the printer connection cable.



3. Turning the power switch to the ON Position

Turn the printer ON.

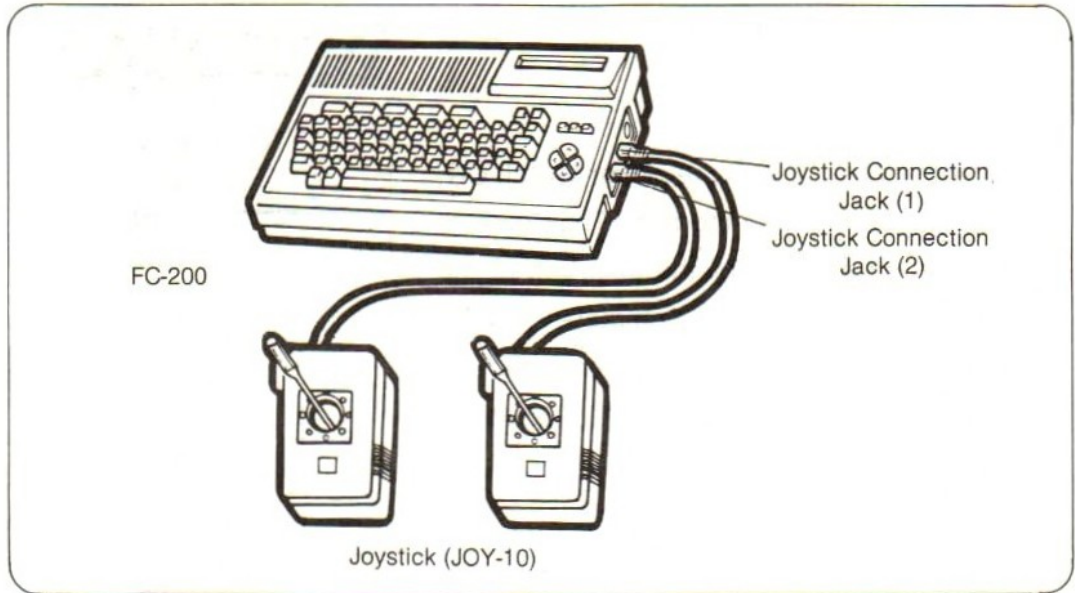
Refer to the printer operations manual.

NOTE

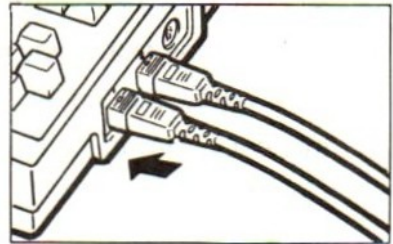
- o The printer INTERFACE connector of the FC-200 is the AMPHNOL 14 pin type. Be sure to use the same type of the interface connector.
- o In case of using the bad connector, miss operation can be caused by the noise and so on.

7-5. CONNECTING THE JOYSTICK TO THE SYSTEM

Connect it to the JOYSTICK CONNECTION JACK (1) on the right side of the FC-200. If two users want to operate, connect them to the JOYSTICK CONNECTION JACK (1) and (2).



* Connect the JOYSTICK CONNECTION JACK on the right side of the FC-200 directly.

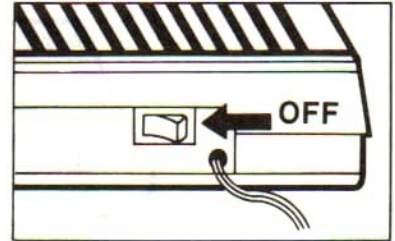


7-6. USING THE ROM CARTRIDGE

You should purchase the ROM cartridge which is sold separately. And insert it into the cartridge slot in front of the FC-200. You should not use the cartridge that does not have the mark of MSX.

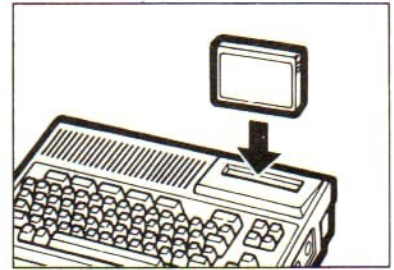
1. How to Insert the Cartridge (The same goes for the game pack)

If you connect it with the power switch ON, the power is out temporarily. The power is turned on again after the insertion of the cartridge into the slot completely (AUTO RESET function). It is recommended to put the cartridge into the slot with the power switch OFF.



- 2) Inserting the cartridge

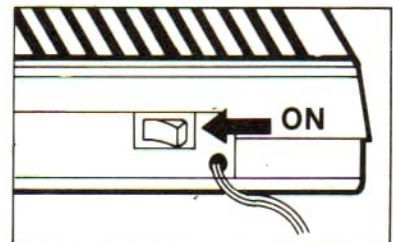
Put the cartridge into the slot with the side labeled MSX backward. Since there is a cover on the slot, if you push the cartridge downward, the cover goes down automatically. The cartridge insertion is completed with a click.



- 3) Turning the power switch to the ON Position

Turn the system ON.

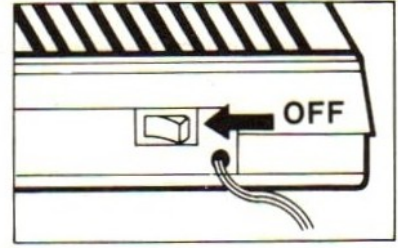
Since the operations of each cartridge is different, refer to the cartridge operations manual.



2. How to Remove the Cartridge

1) Turning to Power Switch to the OFF Position

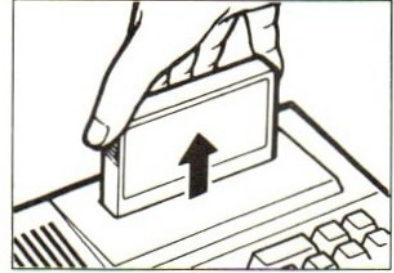
Reach behind the system and turn the ON/OFF Power switch to the OFF Position. If you remove the cartridge with the system ON, the power is out temporarily. The power is turned on again after removing the cartridge from the system completely.



If possible, get rid of the cartridge with the system OFF.

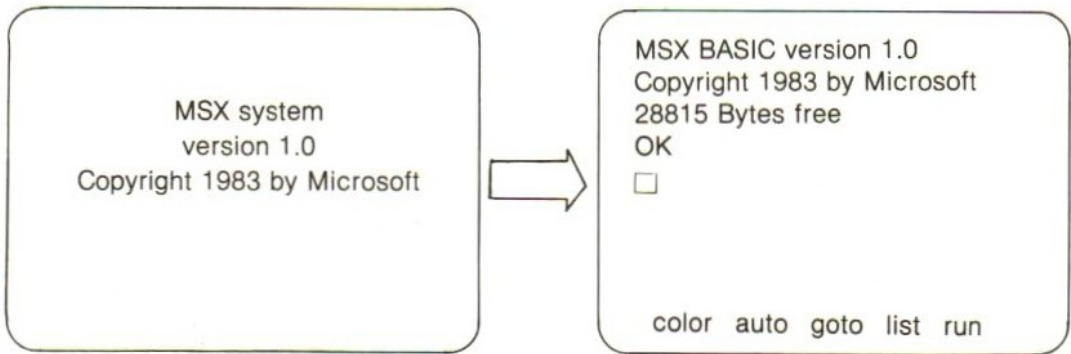
-2) Removing Cartridge

After grasping the side of the cartridge, pull it up steadily without stop.



8. THE ORDER OF OPERATING THE FC-200

1. First confirm the connection of each peripheral device.
Turn the system and the connected peripherals OFF.
2. Turn the peripherals ON.
(Turn the unused peripherals OFF)
Turn the connected peripherals ON.
3. Turn the system ON.
If the system is ON, "MSX System" will appear on the screen for about 4 seconds and then the second message will be as follows:



When the above-messages are not displayed on the screen, T.V screen is not held vertically, turn ON and OFF the system again.

If the message is not displayed normally on the screen after that, refer to the article of "Before asking for service."

NOTE

When the ROM cartridge or Expansion box is connected to the system, T.V. screen is different from cartridge to cartridge.

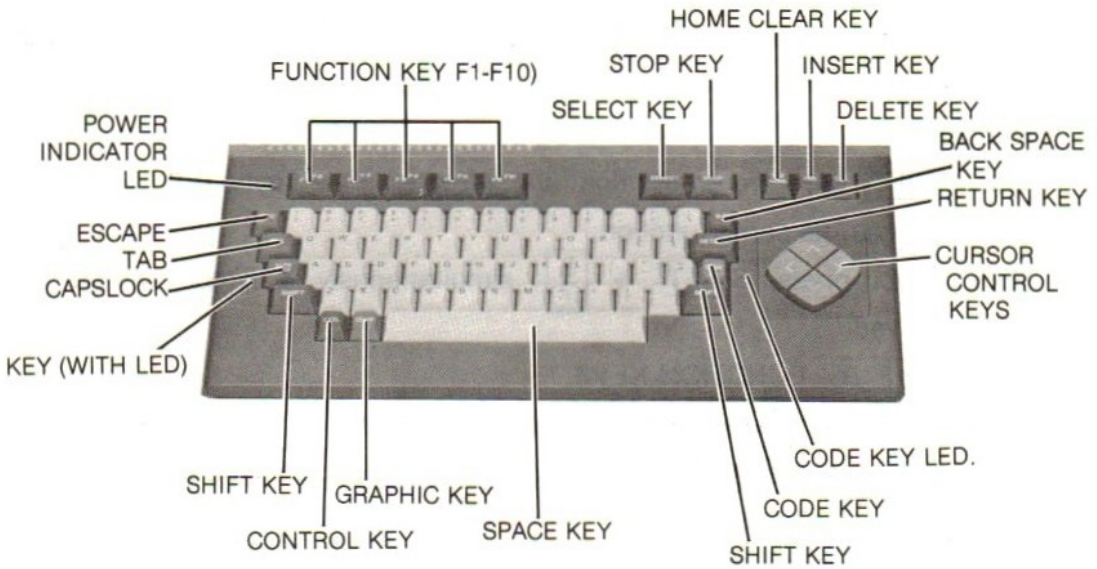
When you first confirm the operation, get the ROM cartridge out of the slot and turn OFF the peripherals.

4. When the above messages are displayed, the FC-200 is ready to receive your program and the program comes to be available.
There are several methods to input the program as follows;
 - 1) You can input the program by the keyboard.
 - 2) The program is input by the CASSETTE TAPE.
 - 3) And also the program can be input by the ROM cartridge.

*When you input the program, the key board is usually used.

9. THE KEY NAMES AND OPERATING METHODS OF KEYBOARD

9-1. KEYBOARD STYLE LAYOUT



NOTE

`ESC` (ESCAPE) key and `SELECT` key are ignored usually. But if you write the program using these keys, they are available.

* SPECIAL KEY (`FUNCTION` , `SHIFT` , `CAPS` , `GRAPH` , etc.)

If you type characters or symbols of the FC-200 respectively, a lot of tedious key-in's are required and it is much more inconvenient.

But, you can do it at once by pressing one key and another simultaneously. Special keys have the function to input many characters with one key and to double key performance in combination with other keys.

9-2. EXPLANATION OF EACH KEY

1. **SHIFT** KEY

By using **SHIFT** key, together with a normal key you can enter the alternate characters or symbols with one key.

But, you cannot enter characters or symbols only with this key.

If you press two keys, **SHIFT** key and another key, simultaneously, the capital characters and symbols on the top part of the keys will be displayed on the screen.

The **SHIFT** keys are on the right and left side of the keyboard.

Each key has the same function.

(Ex.) If you press **SHIFT** + **&7** simultaneously, & symbol will be displayed

2. **F1** — **F10** FUNCTION KEYS

The function keys have pre-defined commands and they are displayed on the bottom of the screen. The command of each key is displayed on the screen with one touch of each key.

(Ex.) Entering **C O L O R SPACE** separately is replaced by one

The following initial commands of the function keys will be displayed on the screen with the system ON.

```
F1 F2 F3 F4 F5  
color auto goto list run
```

If you press the function key with **SHIFT** key the following commands will be displayed.

```
F6 F7 F8 F9 F10  
color cloud"cont list. run
```

NOTE

o The initial commands of the function keys can be re-defined with KEY statement. (refer to MSX BASIC reference for details)

o The range of characters defined by a function key is up to 5 characters on the bottom line on the screen.

(Ex.) **15, 4, 4** will not be displayed by **F6** key.

o Some function keys have the **RETURN** key function with character display.

(Ex.) **F6** key is defined as "color 15, 4, 4 **RETURN**

3. CONTROL KEY CTRL

If you press a certain key with the CTRL key simultaneously, some control operations will be performed.

Refer to CONTROL CODE TABLE in the "appendix" regarding the type of control operation.



(Ex.) If you press the keys CTRL + G simultaneously, "Beep" sound results.

4. TAB KEY

TAB moves cursor to the next tab stop

Tab stops occur every 8 characters.

But Tab stops are different according to the position of the cursor as follows:

(Ex.) 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 cursor is located at left end.
 cursor moves from 1 to 9

(Ex.) 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 cursor is located on "4"
 cursor moves from 4 to 9

5. RETURN KEY

If you press the return key, the symbols and characters displayed on the screen are saved to the system memory of the FC-200.

<p>Explanation of terms: CURSOR</p> <p>CURSOR is "□", a white block, which is displayed with the system ON.</p> <p>If you press keys, characters or symbols will be displayed on the cursor position.</p>	<p>MSX BASIC versio Copyright 1983 b 28815 Bytes free Ok □ ← CURSOR</p>
--	---

9-3. INPUTING CHARACTERS, SYMBOLS, OR GRAPHICS

Using KDB, you can enter alphanumerics and graphic symbols.

To enter a certain type of characters or symbols, you can select **SHIFT**, **CAPS** or **GRAPH** key as you wish.

1. Inputing Alphanumerics and Symbols

1) Inputing following small letters, numerics and symbols

In the state of the **CAPS LOCK** key "OFF" following characters and symbols are displayed.



2) Inputing the following Capital Letters and Symbols

In the shift key "IN" state, you can enter the following characters and symbols



3) Inputting Capitals continuously

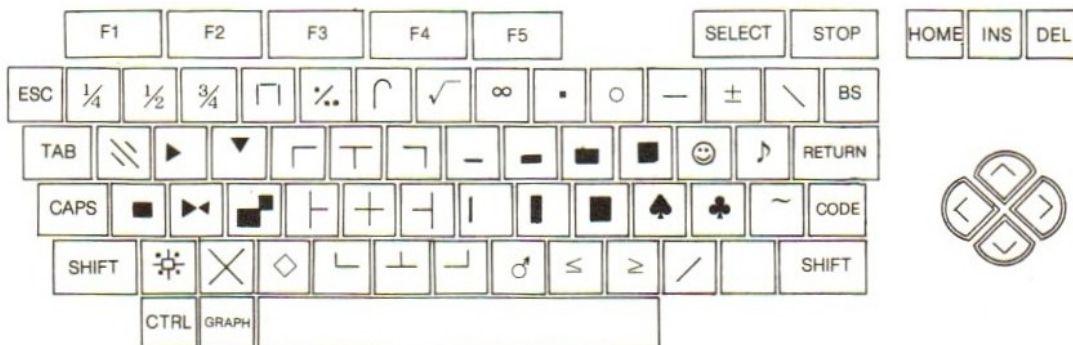
If you press the **CAPS** key once and the LED of the **CAPS** key is turned on, then you can input capitals without pressing the **CAPS** key.

Numbers are returned normally and keys with two symbols produce the lower symbol.

With key activated, you can get lowercase letters with the **SHIFT** key.

2. Inputting Graphic Symbols

If you press the keys with the graphic symbols and **GRAPH** key simultaneously, the following graphic symbols will be displayed on the screen. But, in case of pressing keys without graphic symbols, nothing will be displayed.

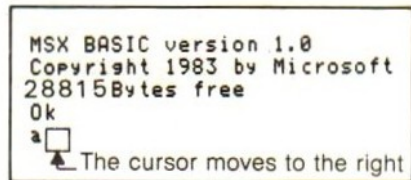


9-4. GETTING USED TO THE KEYBOARD

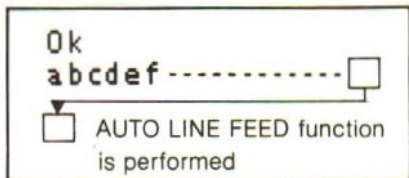
1. Displaying Characters and Symbols

Now the cursor is located under OK.

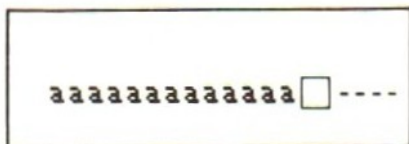
- 1) When you press the key you want to enter (for example a), the character "a" will be displayed on the current cursor position and the cursor moves one space to the right.



- 2) When you continue to press the keys and input data to the end of the line, the cursor moves to the beginning of the next line automatically (AUTO LINE FEED).



- 3) And if you continue to press the same key, the same character will be displayed continuously until you detach your hands away from the key (AUTO REPEAT FUNCTION). Of course, if the character is displayed to the last column of the line, it will be displayed continuously on the next line by AUTO LINE FEED function.



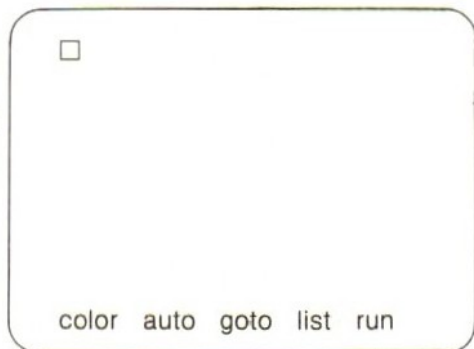
2. Deleting all the Characters Displayed on the Screen

Now, there are characters displayed on the screen. When you want to delete them, press the SHIFT + CLR HOME key simultaneously.

The cursor is located on the top left corner (that is called "HOME POSITION") and all the characters are deleted except the commands of function keys.



SHIFT
+
 CLR HOME



3. EDITING THE SCREEN

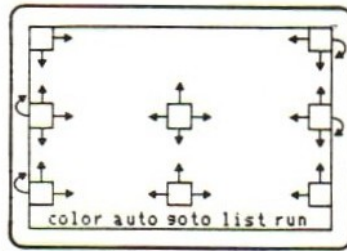
If a typing mistake is made, it is possible to correct the error by first moving the cursor over the wrong character and then typing the correct character.

1) How to move the cursor

The cursor is moved by using the four arrowed keys at the right hand side of the keyboard



When depressed the cursor moves in the direction of the arrows, one touch of a cursor key will move the cursor in the direction of the arrow by 1 character



Pressing two keys simultaneously will cause the cursor to move in a diagonal direction although this facility is of little use except for games playing.

NOTE

- When you use the "WIDTH" command to distinguish a screen, the cursor only moves within that range (refer to MSX BASIC reference for details about WIDTH statement)
- The current commands of the function keys are displayed on the bottom line of the screen, it is not allowed to move the cursor to that line.
- When the

CLS
HOME

 key is pressed, the cursor moves to the top left corner.

2) **Moving the cursor**

This is hardly applied in the process of correcting the program, but the cursor can move in the diagonal directions especially for the games.

When the up-arrow key is pressed with the left-arrow key the cursor is moving in the diagonal direction, to the left.

3) How to insert, delete, and change characters and symbols.

	OPERATING	FUNCTION	DISPLAY EXAMPLE
INSERTION	<p>INS ⊕ (or CTRL + R)</p> <ul style="list-style-type: none"> Pressing INS key a second time will cancel the insertion feature. When you move the cursor by the cursor control key, insertion will be canceled. 	<p>New characters are inserted between the characters already have been displayed screen.</p> <p>The size of the cursor is reduced in the insertion mode.</p> <p>And the characters to the right will be moved as a new character is inserted at the current cursor position</p>	<ul style="list-style-type: none"> 1 2 3 5 6 7 □ ↑ ↓ to insert 4 1 2 3 5 6 7 ↑ ↓ move the cursor Press the INS once 1 2 3 4 5 6 7 ↑ ↓ the cursor is reduced 1 2 3 4 5 6 7 ↑ press 4 key
CORRECTION	<p>Move the cursor to the position of the character that you want to correct, and press the key that you want to enter.</p>	<p>corrects the character displayed on screen and selected by the cursor,</p>	<ul style="list-style-type: none"> 1 2 3 5 5 6 □ ↑ to correct to 4 1 2 3 5 5 6 ↑ move the cursor 1 2 3 4 5 6 ↑ press 4 key

NOTE

The ⊕ and ⊕ signs of the above table have the meanings as follows;

- A** ⊕: the sign ⊕ means that the **A** key have been performed by pressing the **A** key.

(Ex.) **INS** ⊕ means that pressing the **INS** key, insert mode is ON.

- A** + **B** : The ⊕ sign means that **B** key is pressed with the **A** key pressed.

(Ex.) **CTRL** + **E** means that the **E** key is pressed with the **CTRL** key pressed.

Please be sure to remember the meanings of ⊕, + signs.

OPERATING		FUNCTION	DISPLAY EXAMPLE
DELETION	DEL	Deletes the current character and all characters to the right move one space to the left as each character is deleted	0 1 2 3 4 5 □ ↑ to delete 0 1 2 3 4 5 ↑ ↓ move the cursor press DEL key
	BS	Deletes the character to the left. All characters to the right move one space to the left.	0 1 2 3 4 5 ↓ press the BS key 0 1 3 4 5
		In case that the cursor is located at the first column, it deletes the current characters. And all characters to the right are moved one space to the left.	0 1 2 3 4 5 ↓ press the BS key 1 2 3 4 5
	SPACE	Deletes the current character Other characters do not move)	0 1 2 3 3 4 5 ↓ press SPACE key 1 2 3 4 5
To delete all characters	CTRL + E	Deletes all characters to the right.	10A = 10 : B = 10 ↓ 10A = 10 : □
	CTRL + V	Deletes the current line	10A = 10 : B = 10 ↓ □

10. HOW TO USE THE EXCLUSIVE DATA RECORDER AND HOME CASSETTE TAPE RECORDER (C.M.T.)

The FC-200 can load and save the program. If you turn the system ON, all programs written the FC-200 will be erased. Unless you save the program in to the cassette tape recorder, program you made for yourself will be useless.



We wish you purchase the exclusive data recorder for the FC-200 (FCD-10B) and connect it to the system.

Please refer to the articles for the peripheral interface when you connect it to the system.

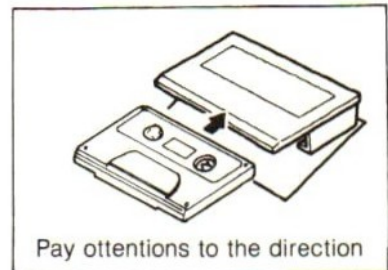
1. SAVING THE PROGRAM

- 1) Put the cassette tape in to the data recorder for the FC-200.

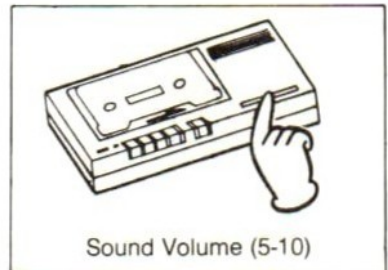
We wish you choose the suitable cassette-tape for the program. Be care about the direction of insertion.

- 2) Have the data recorder set to the recording status.

Before you save the program, adjust the sound volume to the level of 5-10. The numeric values are marked on the volume port of the data recorder. After that, press the "LOAD" and "SAVE" button on the data recorder simultaneously. (In case of the home cassette-tape recorder, press the "RECORD" and "PLAY" buttons instead). In case of the cassette tape recorder with no "REM" terminal, this operation must be executed after the operation of 3).



Pay attentions to the direction

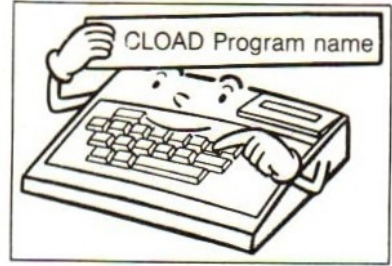


Sound Volume (5-10)

3) Give the statement

C S A V E " program name "

RETURN Since the program name is saved to the cassette tape, give the easy name. (Be careful that the program name is given within 6 characters)



NOTE

Keep in mind that misoperation might happen due to the adjustment of the sound volume of the data recorder. (refer to the BASIC reference for details.)

Explanation of terms CLOAD and CSAVE

"CLOAD" means "loading a BASIC program file from the cassette tape recorder." and "CSAVE" means "saving a BASIC program file into the cassette tape. Another way to load and save the program through the cassette tape recorder is "BLOAD and BSAVE."

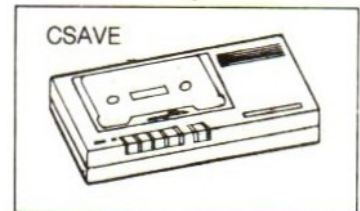
(Ex.) Saving the program name "MSX"

C S A V E " M S X "

The program name.

Type keys in the combination with the shift key simultaneously because they are capital letters.

Press "2" key with the SHIFT key pressed.

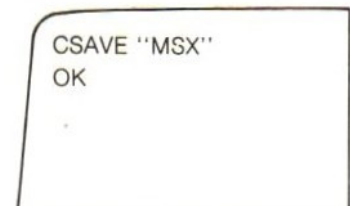


4) Press the RETURN key

Then the cassette tape recorder just starts running with a click and the program begins to be saved in to the cassette tape.

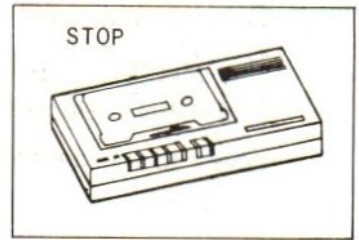
5) When saving the program is over.

The prompt "OK" will be displayed on the



screen, and the data recorder will stop automatically.

(In case of the home cassette tape recorder with no "REM" terminal, you must push the "STOP" button.

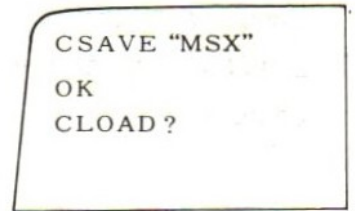


6) Verifying the saved program

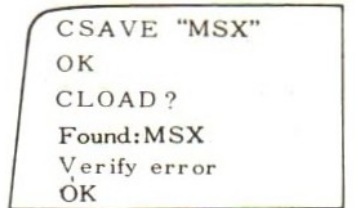
In order to verify the program which is saved in the cassette tape recorder. The following manipulations are required.

(1) Rewind the cassette tape to the point where the program started being saved.

(2) Be sure to enter **C L O A D ?** by typing keys.

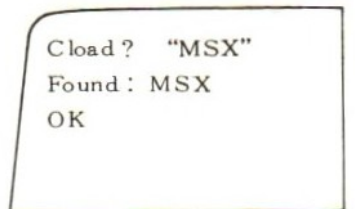


(3) Push the "LOAD" button of the exclusive data recorder. (Push the "PLAY" button for the home cassette tape recorder. In case of the general cassette tape with no REM terminal, push the "PLAY" button after pressing the **RETURN** key.)



(4) Press the **RETURN** key.

Then the cassette tape recorder starts running the program in the FC-200 is compared with the ones saved to the cassette tape in order and in details.



(5) When the program was saved exactly, the prompt "OK" will be displayed on the screen. If the program saved to the cassette tape is different from the one in the FC-200 the message, "Verify error", will be displayed on the screen.

In this case, please do save the program once more.

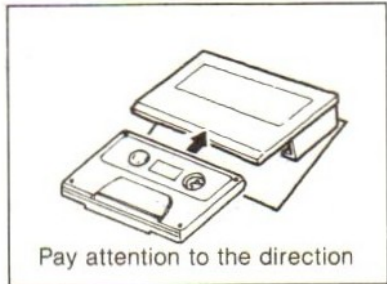
(Ex.) In case of saving and checking the program named "MSX". (See the above feature)

Expansion of terms REM terminal.

REM terminal is designed for the computer to control the STAR/STOP function of the cassette tape recorder when the program is loaded from or saved to the cassette tape recorder. For the cassette tape recorder with no REM terminal, you must manipulate it on your own without the help of the computer. And, the cassette tape recorder is so convenient because the fast winding and rewinding operations are allowed on the condition of plugging the cord into the REM terminal.

2. IN CASE OF LOADING A BASIC PROGRAM FILE.

1) Insert the program tape which you want to load into the data recorder and rewind the tape to the initial state (push the "REW" button)



2) Let the data recorder be in the loading state. Adjust the sound volume to the level of 5-10. Scales are marked on the volume knob. After that, push the "LOAD" button of the data recorder (In case of the general home cassette tape recorder, push the "PLAY" button. When using no REM terminal, Do the step (3) before pushing the "PLAY" button).



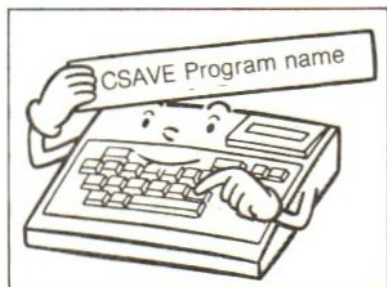
3) Give the statement

Input the following statements with the keyboard.

The **F7** key the type the **program** name.

You can enter **C L O A D** through the alphanumeric key board.

In case of the demo-tape, program name is printed on the label of the tape.



NOTE

When you input the program name, type it exactly (Blank characters must be maintained). And capital letters and small letters can be entered without any changes.

- 4) Press the **RETURN** key.

In case of the general cassette tape recorder with no "REM" terminal, push the "PLAY" button after pressing the **RETURN** key.

The sound of "click" is generated at the moment the **RETURN** key is pressed, and the cassette tape recorder starts running.

And the FC-200 starts searching for the target program among several programs saved to the cassette tape recorder.

- 5) When the FC-200 find out the target program.

If the target program is found out, the message "FOUND; Program Name" will be displayed on the screen. And if it fails to find out the target program, the message "Skip; Program Name" will be displayed on the screen. In that case, Do it again from the step (3) after confirming the program name and the one on the label of the tape. If you finish loading the target program from the cassette tape recorder, the prompt "OK" will be display on the screen. And the exclusive data recorder will stop automatically. (In case of the general cassette tape recorder with no

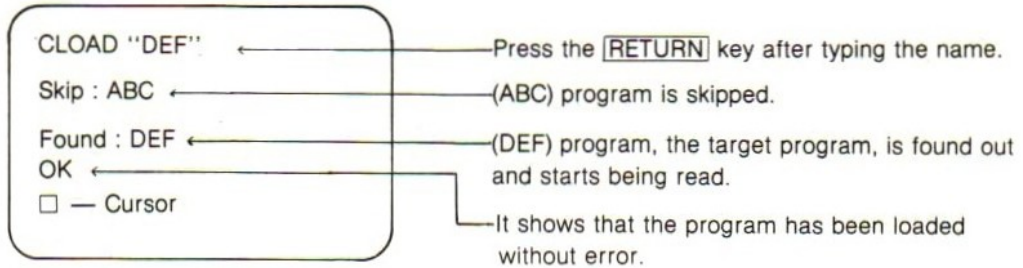
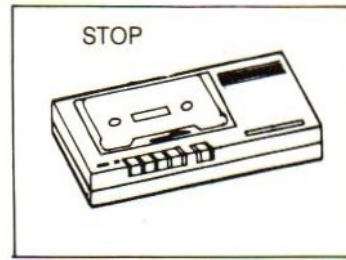
CLOAD "Program name"
Found : Program name

CLOAD "Program name"
Skip : Program name

CLOAD "Program name"
Found : Program name
OK

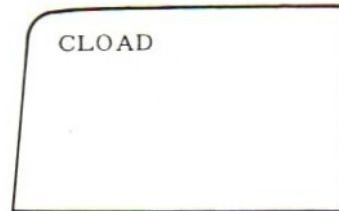
"REM" terminal, you must push the "STOP" button.

(Ex.) (ABC) program and (DEF) program are saved to the data recorder in good order. When you want to load (DEF) program, the following message will be displayed on the screen.



6) In case that you don't know the name of the target program, following procedure will be available.

Type the [F7] [BS] [RETURN] or [C] [L] [O] [A] [D] [RETURN] key, the [F2] [BS] [RETURN] keys with the [SHIFT] key pressed. (When you do not name the program, don't enter the " " symbol after CLOAD). Then the FC-200 will load the program found first among several programs that had been saved in the cassette tape recorder.



If you continue "CLOAD" in order, you can load the programs from the cassette tape recorder one by one to the last.

7) Holding the operation in the process of loading.

When you can not find out the target program by "CLOAD" and you want to stop loading, press the [CTRL] + [STOP] keys simultaneously. Then "DEVICE I/O Error" will be displayed on the screen and loading the program from the cassette tape recorder will be stopped. After adjusting the

data recorder again, type the exact program name and load the program from the cassette tape recorder again.

NOTE

- The FC-200 is designed for being operated exactly according to the wide level of the sound volume and tone of the general cassette tape recorder. But if the sound volume is too much small or the tone is too bad, misoperation will be caused. In that case, adjust the sound volume knob and tune the knob to the proper location between the mild — level and the maximum — level through testing CLOAD.
- In case of the cassette tape recorder with high power output, adjust sound volume between the minimum level and the mid — level, and locate it in the most proper position.

11. HOW TO ADJUST THE MONITOR AND THE COLOR T.V.

The color quality that the FC-200 creates may be changed according to the controlling state of T.V. or Monitor.

With the system ON, the background color is dark blue in its initial state. And the color of the characters is white. Before you operate the system, please be sure to adjust the initial color. (refer to the T.V. operations manual for details)

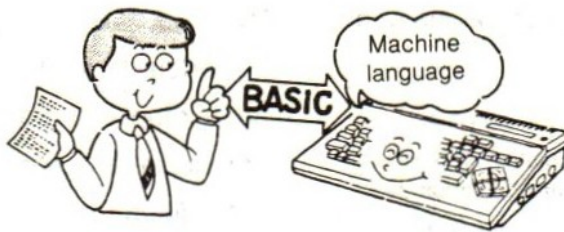
12. HOW TO MAKE THE PROGRAM

The FC-200 operates through MSX BASIC language. Make good use of MSX BASIC language after reading this chapter sufficiently.


12-1. WHAT IS BASIC?

Computer is operating through the combination of binary 0 and 1 (This is called Machine language). It is, however, difficult to use this machine language, so we use BASIC based on the language that we can easily understand—English.

The BASIC is so easy that thereby even beginners can make the program without any difficulty and correct it with ease.



12-2. THE EXECUTION OF STATEMENTS

You can execute a BASIC statement by typing the keys you want and pressing the **RETURN** key (from now on,  symbol means pressing the **RETURN** key). Even if you enter a BASIC statement, it is only displayed on the screen but cannot be executed without pressing the **RETURN** key. Since alphanumeric characters entered through the keyboard are displayed on the screen by the PRINT statement.

Explanation of terms : BASIC

BASIC is an acronym for Beginner's All-purpose Symbolic Instruction Code. This is the program through which beginners can operate the computer by entering the easy English words.


MSX BASIC language

Refer to MSX BASIC Reference, the manual which describes the FC-200 Basic languages in detail. When you make algebraic expressions or enter instructions, Computer will not operate unless you enter them according to the MSX BASIC Reference.

1. Displaying Characters

PRINT statement is used for displaying alphanumeric characters after itself on the screen and expressed by enclosing those characters with quotation marks. The string within quotation marks are composed of capital letters small.

Print "MSX BASIC"	←	enter the command
MSX BASIC	←	the result
OK	←	execution
□	←	the execution is over cursor

For example, when you press the  key after entering PRINT "MSX BASIC", the computer executes it and "MSX BASIC" will be displayed on the screen.

2. Displaying Numeric Values and Calculate

```

?58*9-21 ← calculation of 58*9-21
              (input of command)

501 ← The result of the
      calculation
Ok
      calculation of
?36+9/3 ← '36+9+3'
              (input command)

39
Ok
      calculation of
?int(11/6) ← '11+6'
              (input command)

1 ← Quotient is truncated into an
   integer
Ok
  
```

In addition to displaying alphabetic or symbolic characters, it is possible to display the numerics and make a calculation by the PRINT statement. A question mark can be equally used in place of the PRINT statement. Don't forget to press the **RETURN** key after entering the statement.

If you type `[?][5][8][*][9][−][2][1]`, the result, 501, will be displayed on the screen.

We are using `[*]`, `[/]` in place of `[X]`, `[÷]` as in the above figure. We call `[*]` as Asterisk and `[/]` as slash.

3. Executing the Statement Again

When you want the system to re-execute the statement executed by the **RETURN** key you must input the statement again. So if you input the statement again or move the cursor to the position of the command, and press the key, the statement will be executed again. And it's possible to change some portions of the previous expression.

Locate the cursor position to "5", change "5" to "6", and press the **RETURN** key, then the calculation of $100-2^6$ will be executed.

Executing BASIC command by pressing only the **RETURN** key like this is possible only for simple expressions or calculations.

To execute more complicated calculations, they should be executed through the orderly combination of several commands.

The procedure that many commands are executed like this is called a program.

```

[?] 100-2^5 ← calculation
              (input of command)

68
Ok

□ ← move the cursor to [?]
   position and press the
   [RETURN] key

? 100-2^5 [?] ← enter the key
                from the first
  
```


12-3. DIRECT MODE AND PROGRAM MODE.

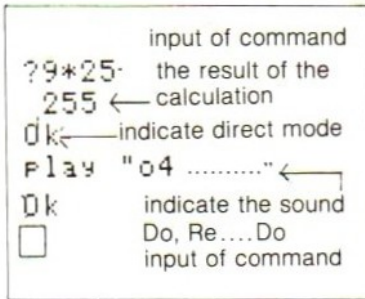
When you turn the FC-200 ON, the initial message will be displayed. After the prompt "OK" has been displayed, the characters or symbols you enter will be displayed.

We call it "direct mode" that BASIC commands are executed the moment they are entered, and "program mode" that program is executed.

1. Direct Mode

DIRECT MODE is available for following two operations

- 1) Executing a command. For example, as soon as you enter the commands and press the RETURN key, the computer executes them like in the example.



And the computer is waiting for the next key input. At that time, the prompt "OK" which means that it is ready to accept new commands will be displayed. As mentioned above, since the commands that were executed earlier are not stored in the memory,

(Pressing the **SHIFT** + **CLS HOME** keys, can erase the screen.)

You must re-type the commands if you want to execute the commands which previously used in direct mode.

Explanation of terms COMMAND and STATEMENT

There is no exact difference between commands and statements; commands are used in DIRECT MODE without line numbers, but statements are sets of commands.

2) How to Edit and Correct the Program

BASIC statements allow the user to enter the line number from 1 to 65529. Line numbers indicate the procedure that the program is executed.

For example, even though you enter **10 PRINT "MSX"**, "MSX" is not displayed on the screen. But it is stored in the memory. Even if you type the following keys, all of them are only stored in the memory but not executed.

20 PRINT "BASIC"
30 ? 10 * 8 - 5

First, clear the screen. (Which does not mean that the memory is cleared.) And then, the program on the right will be displayed on the screen when you press the **LIST** and **↵** keys.

```
list
10 PRINT "MSX"
20 PRINT "BASIC"
30 PRINT 10*8-5
Ok
```



To display the statements (the program) saved in the memory is said to LIST.

The program, even though cleared on the screen, can be listed on the screen because it is still stored in the memory. And it is possible for you to edit the contents of the program as you wish.

2. Program Mode

DIRECT MODE is converted to PROGRAM MODE by typing the **R** **U** **N** and **R** **E** **T** **U** **R** **N** keys. And the conversion from PROGRAM MODE to DIRECT MODE are made as follows;

- 1) When there is no line number to be executed after executing the program in order or when the program execution is over with the **E** **N** **D** statement.
- 2) When the program executions are suspended by the **CTRL** + **STOP** keys (The line number where the program execution is suspended is indicated the message "Break in —")
- 3) When there is an error code or an error message in the program (error messages are displayed as — error, etc)

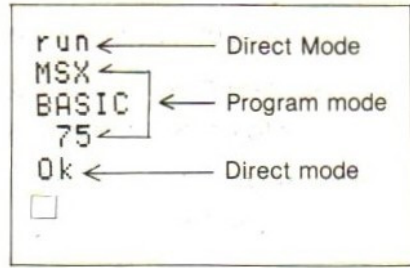
Explanation of Terms PRINT

The question mark (?) can be used equally in place of the PRINT statement. In case of listing the program, question mark (?) is replaced by "PRINT" on the screen.

But, if you enter "L?", it will not be displayed as "LPRINT" (It is not accepted as BASIC language)

Key input is not accepted in the process of executing the program as long as it is not especially specified.

Execute the foregoing program by pressing the **RUN** and **RETURN** keys.



JUST A MOMENT!

- The flow of key input

The operation of the computer is controlled by L.S.I chip called "CPU".

The characters and symbols entered through the KBD are not only stored in the video display memory and displayed on the T.V. screen, but also are stored in the direct mode memory. When you press the **RETURN** key, CPU executes the commands in direct mode memory. When the line number is followed by the statement entered by the KBD.

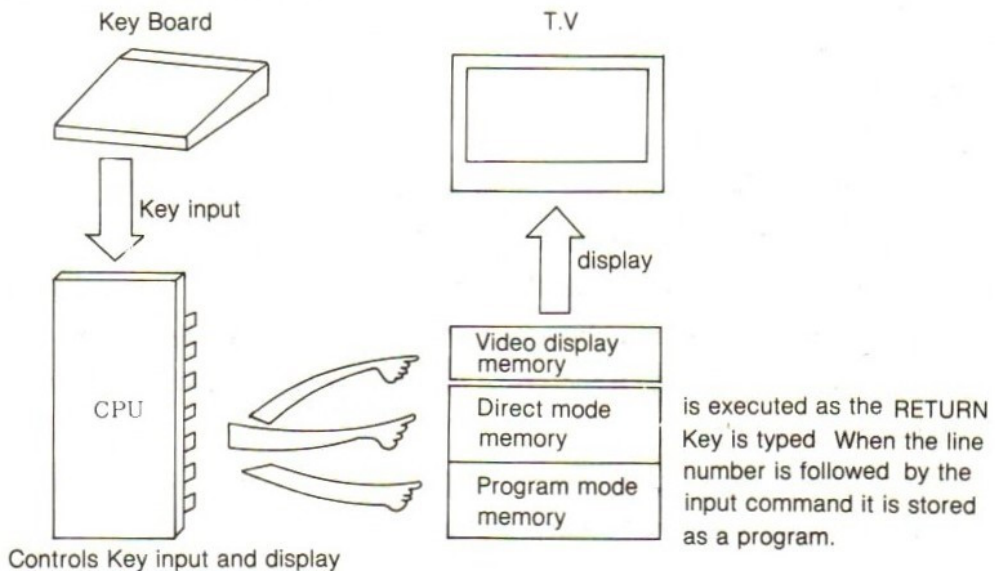
The program line is stored as a part of program in the program mode memory.

The program starts running as the RUN command is stored in the direct mode memory and executed. And when the program stops running or errors are generated. It returns to the direct mode.

Since CPU knows the contents of the video display memory and the position of the cursor, if you move the cursor to the position of the command which was executed once and press the key, then it executes that command.

CPU

(Central Processing Unit): As the most important Integrated Circuit chip in the computer, it controls how to execute the command when a command is entered into the computer.



12-4. HOW TO CORRECT THE PROGRAM

It is very difficult to write a perfect at the beginning. We often can correct the errors generated by the wrong key input and edit a program into the better one. Since how to correct the errors by the wrong key input was explained previously, we are explaining how to edit programs here.

1. Inserting of Program Line

Since the program is executed in order from the smallest line number, in case of adding a statement, insert its line number where you want the additional statement to be executed. For example, when you want to insert the following

command between lines 20 and 30 in the right figure;

```
list
10 PRINT "MSX"
20 PRINT "BASIC"
30 PRINT 10*8-5
Ok
25 ? "10*8-5="; ←addition
 command
```

P R I N T " 1 0 * 8 - 5 = " ;

Type the keys as follows;

2 5 ? " 1 0 * 8 - 5 = " ; ↵

and press the **LIST** and **↵** keys after clear-screen (**SHIFT** + **CLS HOME**)

Then line No. 25 will be sure to be placed between the line No. 20 and line No. 30.

2. How to Convert the Current Statement into another one

When you want to convert a statement into another one, assign the same line number to the statement that you want. When the commands of the same line number are entered, the command entered later has priority in BASIC.

```
list
10 PRINT "ABC"
20 PRINT "BASIC"
25 PRINT "10*8-5";
30 PRINT 10*8-5
Ok

```

For example.

press the **LIST** and keys after entering **1 0 P R I N T " A B C " ↵** in the above program.

Then the command in the line No. 10, will be converted like in the above figure. In another way of conversion, move the cursor to the line No. 10 and convert "MSX" to **A B C** and press the **RETURN** key, then you can get the converted program.

3. How to Delete the Program Line

You can delete the unnecessary line as follows;

1) When you want to delete one specific line, Enter the line number in the first place and press the RETURN key,

(Ex.) `25 ↵` — the line No. 25 is deleted

2) When you want to delete the lines in the specific range, Use the DELTE statement.

(Ex.) `DELETE — 20`; deletes the program lines from the first to the 20th

(Ex.) `DELETE 20 —`; deletes the program lines from 20th to the last.

3) When you want to delete the entire program

Enter `NEW ↵`

JUST A MOMENT!

There are the following two methods in entering the same statements continuously and repeatedly like in the right figure, in order to input the similar program quickly.

You can quickly enter the program by one way or another.

(1) Make use of the key command

When you enter the following command.; Key 10, "Print" + Chr \$ (&H22) `↵`, and press the `F10` key, PRINT" command is displayed, so it is not necessary for you to enter each character.

(2) Convert the line number and enter it.

Enter `PRINT "M" ↵`, move the cursor to the position you want, and convert it in to `20 PRINT "S" ↵` (10 → 20, M → S)

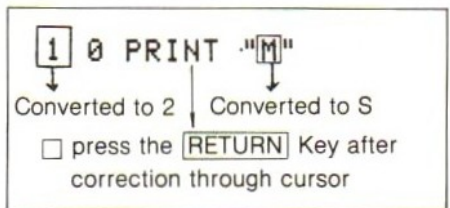
Lines 10 and 20, are entered like this.

But, since this method converts the command displayed on the screen to another one, be careful not to make an error in entering the line No. and command in from now on.

The exception in deleting the program line No.

In case that the line number is displayed automatically by AUTO statement.

```
10 PRINT "M"
20 PRINT "S"
30 PRINT "X"
40 PRINT "B"
50 PRINT "A"
60 PRINT "S"
70 PRINT "I"
80 PRINT "C"
```



* (Asterisk) mark follows the line number if the same line number that has already been entered, is input. At that time, the statement still remains where it is without being deleted despite of pressing the **RETURN** key.

Refer to MSX BASIC REFERENCE FOR DETAILS.

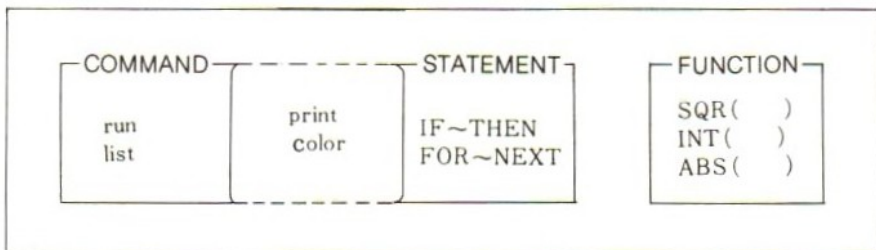
12-5. THE PRE-DEFINED LANGUAGE OF BASIC

We are explaining the languages used frequently and the terms you should know about before practicing the BASIC language program.

1. Pre-Defined Language

Among the languages that order the computer to do this or that, what is used in the Direct mode is called "Command" and what is used in the program mode is called "Statement". But, there is no drawing the line between them since there are too many commands used in the Direct mode and Program mode both.


Since the commands used in MSX Basic and the functions used for calculations are all based on English, those are called "pre-defined languages". The capital and small letters of pre-defined languages are considered as the completely identical ones, but you have to enter characters exactly. For example, PRINT commands is the same as follows; print, Print, PPrint (regardless of blending capital letters and small letters), but PRINT (spacing one character out) will generate an error.



2. The Function

The FUNCTION means what executes some calculations or manipulations about the given-data when you enter some data.

It is not allowed for the FUNCTION to be used alone, but it is always used with the other command.

For example, when you enter INT (3. 14)  () and the command that the value in the bracket be truncated in to INTEGER, the error is generated. But when you enter it with the PRINT command as follows; PRINT

INT (3, 14) —, "3" will be displayed on the screen.

MSX BASIC has the "intrinsic" functions in the system, such as ABS (Absolute value), INT (being truncated in to an INTEGER), SIN (sine function), COS (cosine function) and PEEK. MSX BASIC also allows "User-defined" function that are defined by the Programmer:

"User-defined" functions are used to make the program simple by converting the long statements (the calculation expression) to the short functions when you repeat the same calculation expression many times.

JUST A MOMENT!

About the user-defined function (DEF, FN)

The command that defines and names a function is called "DEF FN" command, and the function defined by this "DEF FN" command is called "User-defined function". Let's take a program for instance that draws ♥ marks on the screen properly like in the right figure.

The "DEF FN" command defines the function name by "FN" and the English characters that follows it (ex: FN A, FN B, FN CD.....) and the function is defined by entering the keys as follows;

```
DEF FN A (A,B) = A*B-B/10
```

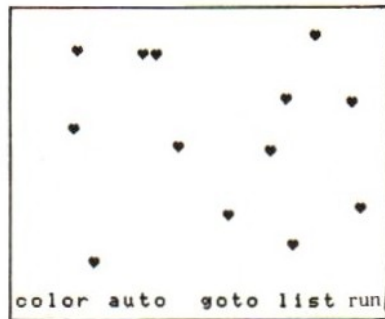
function variable name — calculation expression name

First, enter the program like the figure on the right.

Program is executed by pressing the RUN key (or pressing the **F5** key).

If you want to stop the execution of the program, press the **CTRL** + **STOP** keys.

If you do not use DEF FN command on lines 20 and 30 line statements, are as follows, $X = \text{RND}(1)*30$, $Y = \text{RND}(1)*21$ but the lines 20 and 30 have simple forms by applying the user-defined function to the line 10 and defining as follows; $\text{FNA}(\text{variable}; A) = \text{RND}(1)*\text{Variable A}$.



```
USER-defined function
10 DEF FNA(A)=RND(1)*A
20 X=FNA(30)
30 Y=FNA(21)
40 LOCATE X,Y:PRINT "♥"
50 GOTO 20 ← repeat
```

used functions
mark

(Summary of the program)
In line 10, "user-defined function is defined. RND (1) statement generates a different random number with a value between zero and one. 30 in line 20 sets the horizontal position (X direction) to 0 through 30, and 21 in line 30 the vertical position (Y direction) to 0 through 21.
The position of the cursor is determined in line 40 and " ♥ " mark will be displayed there.
Line 50 means that the program branches to line 20 and the loop is executed.

3. The Constant and the Variable

There are two methods for calculations in the program.


1) PRINT 100-2^5.

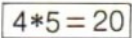


2) A = 100:B = 2:C = 5:PRINT A-B^C



Since 1) and 2) methods have the same result, the value of 68 will be displayed on the screen when the program is executed. But 1) method is called "calculation through constants" and 2) method is called "calculation through variables." After all, the result of the first method is always constant but the value of the second method will be variable according to the manipulation of the program.



For example, when you enter the two numeric values and multiply them, the result is as follows;

Enter the program like the figure on the right.

Since ? is displayed on the screen by the RUN and  keys 4*5 is calculated and

 is displayed when you enter  

  after "?". And "?" is displayed on the screen and waiting for some values to be entered.

(The program is composed of Loops, press the  +  keys to hold it)

Constants mean the fixed numeric values, but * or = is also called a string constant.

Just as there are "string variables" and We are explaining them in the next paragraph.

```
10 INPUT A,B
    → input 2 numeric values
20 PRINT A;"*";B:
    → shows variables
30 PRINT "=";A*B
    → shows the result of
40 GOTO 10
    → repeat calculation
```

Explanation of terms Constants and Variables

Generally speaking, we call the string-typed constants "string" and the string-typed variables "the string variables"

And the numeric constants are simply called "constants" and the numeric variables are called "variables"

- String-typed constants and string-typed variables.

We call the constants with the fixed contents as "MSX" in the figure on the right

```
10 PRINT "MSX"
20 A$="BASIC"
30 PRINT A$
```

“string” (string-typed constants). And variables having the changeable contents like A \$ is called “String variables” (String typed variables).

What is enclosed in “ ” mark means the contents of the string variables. The string variables can be represented by two English characters and ‘\$’ symbol after them.

For example.

A\$, B\$, C\$.....

A1\$, B2\$, C1\$.....

AS\$(1), BC\$(1).....(Refer to MSX BASIC reference for details)

- The numeric constants and the numeric variables.

We call the numeric constants simply “constants” and call the numeric variables “variables”, “A” is a variable and “5” is a constant in the figure on the right.

Among constants and variables, there are the single precision variables and the double precision variables (Refer to MSX BASIC reference for details)

```
10 INPUT A
20 PRINT A*5
```

The default constants and variables is the double precision in the FC-200, as long as it is not expecially specified.

1) Integer type

It is the set of integers within the range of -32768—32767. The numeric followed by “%” is called “integer type”.

(Ex.) 3.14%—3.3168%—3168

converted to Integers when they are followed by %.

(Ex.) A = 5.14—A% = 5.

Explanation of terms Array Variables

A\$(1.1)

(0,0)	(1,0)	(2,0)
(0,1)	(1,1)	(2,1)
(0,2)	(1,2)	(2,2)

An array is a group or a table of values reference by the same variable name. Each element in an array is referenced by an array variable that is subscripted with an integer or an integer expression. Array variable names have as many subscripts as there are dimensions in the array, (from the first dimension to the 255th dimension)

2) The single precision

We call real number followed by "f" symbol the single precision.

It's significant numerics are six digits and the number above six digits is rounded off at the seventh digit.

(Ex.) 100f—100 (100 is displayed on the screen)

12345678f—123456

12345678f—123.456 A! (a single precision variable)

2) The double precision

Real number above 7 digits or followed by "d" are called "the double precision". If significant numerics are 14 digits and the number above 14 digits is rounded off at the 15th digit.

(Ex.) 3.14159265358979323

— 3.1415926535898

314159265358979323 —

3.1415926535898 E + 17

↑
Exponential

form

A (The double precision variable)

Integers and variables without "%", "f", "d" symbols are considered as double precision ones.

JUST A MOMENT!

Binary numbers, octal numbers, decimal numbers, and hexadecimal numbers. There are binary, octal, and hexadecimal numbers besides decimal numbers that we are usually using to denote the numeric values.

Those are not too much often used, but they are available to display the data and have the computer make calculations.

Refer to the table, and make the best use of it.

When you use them as the data of computers they are as follows;

1101 (binary number), 015 (octal number)

13 (decimal number), D (hexadecimal number)

(All are 13 in the decimal number)

Binary	Octal	Decimal	Hex.
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F
10000	20	16	10

& B 1101 & O15 13 & HD

12-6. THE BASIC OPERATION OF PROGRAMMING

We think you have already understood how the program is written and edited line numbers, commands, and statements.

The foregoing explanations of the terms are basic for editing the program. Refer to the foregoing explanations once in a while during reading what will be explained about how to run programs from now on.

We are explaining the necessary key manipulations and program terms in writing the program in this section.

1. THE LINE NUMBER AUTOMATICALLY (AUTO STATEMENT)

Every BASIC program line always begins with line number.

Since the line number is generated automatically by the AUTO statement, it is not necessary to enter the line number by pressing keys.

1) Enter the AUTO statement

AUTO statement is entered as follows; AUTO ((the initial line number)) ((increment)). You can omit the contents in the square bracket.

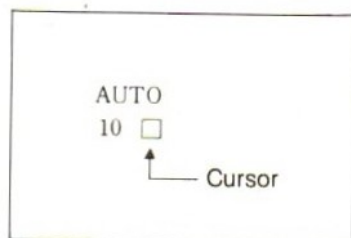
But at this time, the default for (the initial line number) and (the increment) is 10, automatically.

Enter the followings;

A **U** **T** **O** (**F2** **↵** is O.K)

Then the line No. 10 will be displayed on the screen like the figure on the right.

When you enter the command in the line 10 and then press the **RETURN** key, the next line number 20 will be displayed.



2) The AUTO statement is terminated by typing the **CTRL** + **STOP** keys.

In case of terminating the AUTO statement, the last line number will be displayed, but that line is not in the program.

The **CTRL** + **C** keys have the same operation as that of the **CTRL** + **STOP** keys.

NOTE

- If the AUTO statement generates a line number that was already used, an asterisk (*) is printed right behind the line number to warn the user to replace the existing line with any input.

However, pressing only the RETURN key will not erase the line and the next line number will be displayed (Refer to MSX BASIC reference for details.)

- When the computer is waiting for data input (In the process of executing AUTO statement and INPUT statement) the computer operation can be terminated by typing the **CTRL** + **C** keys.

And when the computer is not only waiting for data input, but also in the process of executing the program, typing the **CTRL** + **C** key will (also) terminate the operation.

1. KEY INPUT IN THE PROCESS OF THE PROGRAM EXECUTION (INPUT, IN-KEY \$ STATEMENT)

During the program execution, key input is not allowed except the special

keys (for example, the **CTRL** + **STOP** key). But the key input is allowed in the course of programing.

1) INPUT statement

When an INPUT statement is encountered, program execution pauses and a question mark is displayed to indicate that the program is waiting for data input. For example, "?" mark will be displayed after entering **RUN** in the right figure. After that, if you enter the numbers from 0 to 24, the same number of clover marks (♣) will be displayed.

Like this, through the INPUT statement, program execution pauses and the program is waiting for data (String 3 numeric values, etc).

INPUT C\$ causes the program to be waiting for string variables.

```

10 Y=Y+1
20 INPUT B
30 IF B>24 THEN 20
40 C#=STRING$(B,"♣")
50 LOCATE 4,Y:PRINT C#
60 GOTO 10
    
```

(Summary of the program)
 In line 20, the program is waiting for data. Here numerics are specified by key input. And if you enter characters, "Redo from start?" will be displayed. In line 30, it is verified that the entered value is less than 24. In line 40, "♣" as number of key input are displayed. In line 50, the position of C\$ is determined. In line 10, the position of C\$ is down by one line for the next input. (Refer to MSX BASIC reference for details about statements)

2) INKEY \$ Statement

This statement enables key inputs to be accepted even in the process of program execution.

Since the contents of INKEY \$ are changed by typing keys, they are usually saved to one of string variables as A\$ = INKEY \$ in the right figure. (Refer to MSX BASIC reference for details)

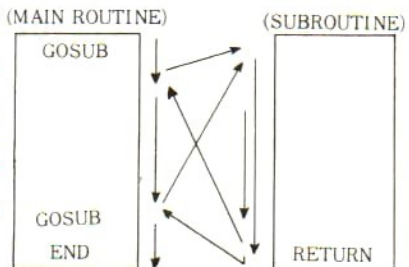
```

10 A$=INKEY$
20 IF A$="" THEN A$=" "
30 PRINT A$;
40 GOTO 10
    
```

(Summary of the program)
 In line 10, key input is assigned to A\$. Line 20 shows that the space key was input if there is no key input.

3. JUMP STATEMENTS (GOTO, GOSUB STATEMENT)

Program execution begins from the smallest line number in order, but use of GOTO statement or GOSUB statement is used to let the program branch unconditionally out of the normal program sequence to a specified line number and execute it.



1) GOTO statement

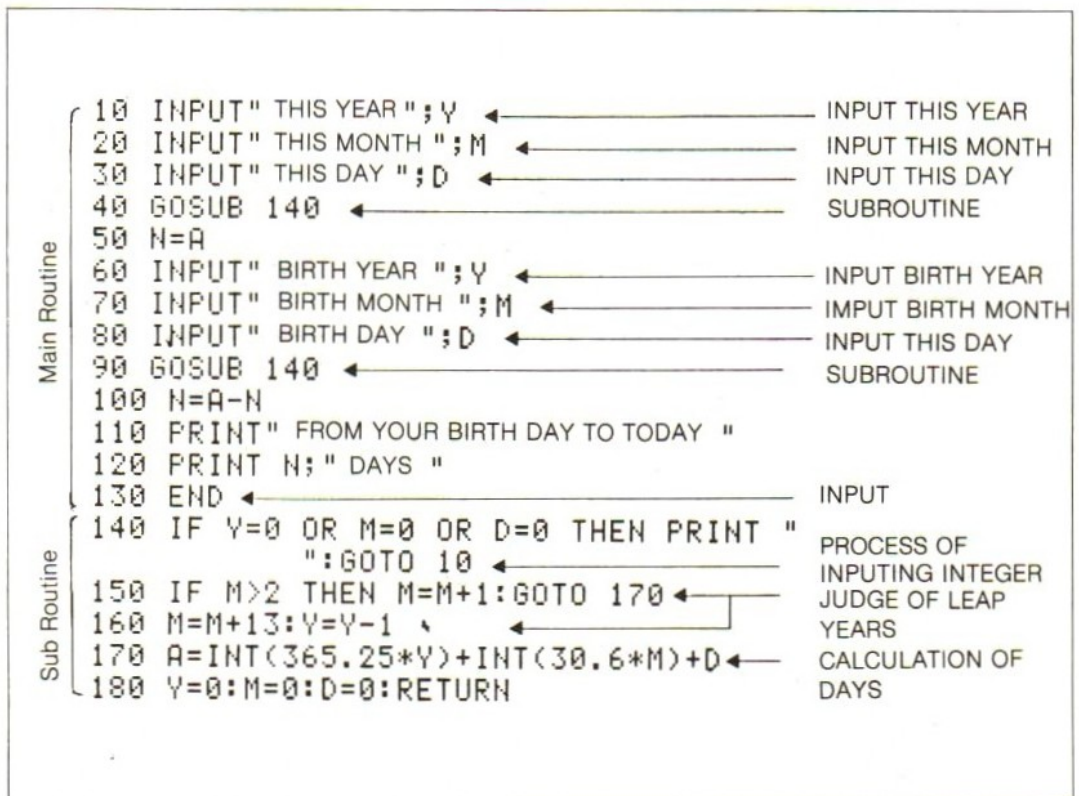
It will let the system branch unconditionally out of the normal program sequence to a specified line number.

2) GOSUB statement

In case of some of the program being executed repeatedly, the use of this statement will make the program shorter.

The parts called by GOSUB statement are called "SUBROUTINE". There is the **RETURN** statement in the last line of the subroutine and it means that the parts of the subroutine is terminated and cause the program to branch back to the statement following the most recent GOSUB statement. But it is recommended that the subroutine be readily distinguishable from the program (what we call-MAIN ROUTINE)

To distinguish from the subroutine, the program is called "MAIN ROUTINE".



4. CONDITIONAL JUMP (IF—THEN)

(IF—THEN, IF—GOTO, IF—THEN—ELSE, ON—GOTO statement)

By the fore going GOTO, GOSUB statements, the system never fails to branch unconditionally to a specified line when the program execution comes to those statements.

But the system always branches conditionally to a specified line by the following statements; IF—THEN, IF—GOTO, IF—THEN—ELSE, etc.

1) IF—THEN

IF <u>Y=0 OR M=0 OR D=0</u>	THEN	<u>PRINT " " : GOTO</u>
conditional expression		Execution after expressions being recognized

If the result of the conditional expression is true (in case of the above example, the values of Y,M,D are 0), the THEN clause is executed.

If the result of the conditional expression is not true (in case of $Y \neq 0$, $Y \neq 0$, and $D \neq 0$), the "THEN" clause is ignored and the system continues the next executable statement.

2) IF—GOTO (In case of conditional JUMP)

<u>IF X>0</u>	<u>GOTO 120 (or IF>0</u>	<u>THEN 120 will be O.K.)</u>
Conditional	Jump command	Jump command

If the result of the conditional expression is true ($X = 0$ in the above example), the system branches to line 120 and then the line is executed.

If the result of the expression is not true, GOTO clause is ignored and the execution continues with the next executable statement.

In this case, THEN (line number) statement can be used in place of GOTO (line number) statement and lead to the same program execution.

3) IF—THEN—ELSE

<u>IF X>0 THEN</u>	<u>PRINT : GOTO 120</u>	<u>ELSE PRINT ABS(X)</u>
Conditional expression	execution in case of being true	Execution in case of not being true

If the result of the conditional expression is true, THEN clause is executed, but if it is not true, ELSE clause is executed.

4) ON—GOTO

ON A GOTO, 100, 110, 120

Parameter list of line number

This statement let the system branch to one of several specified line numbers, depending on the value of the variable (called parameter). The value of parameter determines to which line number in the list will the system branch.

```
10 INPUT B
20 A=INT(B/10)
30 IF A>3 THEN 90
40 ON A GOTO 60,70,80
50 PRINT "10 Less than":GOTO 100
60 PRINT "10 to 19":GOTO 100
70 PRINT "20 to 29":GOTO 100
80 PRINT "30 to 39":GOTO 100
90 PRINT "30 More than"
100 END
```

If the value of the parameter is zero or larger than the line number to which the system branches, it continues with the next executable statement (We are handling the case the value is greater than 3 in the right figure).

JUST A MOMENT!

IF THEN the (conditional expression) is called the logical operation or the relation operation.

The relational operation is expressed by way of the following operators which are used to compare two values; =, <, >, < > .

(Ex.) (Relational operation)

10 < 5	false	If the result of comparison
10 < > 5	true	is not true, it will turn
10 > 5	true	out to be false
10 = 5	false	

Explanation of terms Branch Command

The conditional command in IF—clause is called "Branch Command" and the line that is branched to and executed is called "Branch line."

5. LOOP STATEMENT WITH CONDITIONAL STATEMENT (FOR—NEXT, IF—GOTO statement)

If you **RUN** in the program like in the right figure, the program execution continues

```
10 INPUT A,B
20 PRINT A;"*";B;
30 PRINT "=";A*B
40 GOTO 10
```


as long as **CTRL** + **STOP** is not entered or the error message is not generated.

(Well call this "Infinite Loop")

On the contrary, we call it "definite loop" that a series of instructions are performed repeatedly in a loop in a given number of times.

1) FOR — NEXT statement

When the variables are within the fixed range like the below program, the program is executed repeatedly in the FOR — NEXT loop.

For (Variables) = (initial value) TO (final value) STEP (increment).

The program lines following the FOR statement are executed until the NEXT statement is encountered. Then the variables is incremented by the amount specified by STEP. It is checked if the value of the variable is greater than the final value. If the step is positive, the loop is executed until the initial value is equal to the final value.

If step is not specified the increment is assumed to be one.

The all characters assigned by computer are displayed on the program in the right figure.

```
10 FOR N=&H21 TO &HFF
20 PRINT CHR$(N) ; " ";
30 NEXT N
40 FOR K=&H41 TO &H5F
50 PRINT CHR$(1)+CHR$(K); " ";
60 NEXT K
```

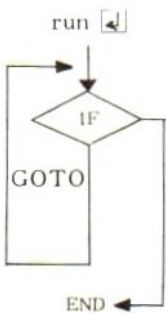
2) IF—GOTO command (As loop statement with the conditional statement)

It is possible to execute the loop repeatedly with IF statement and variables.

In case of using IF statement, there are two(2) methods as follows; (But the operations are nearly identical)

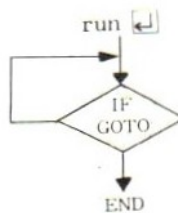
```
10 N=1
20 IF N>15 GOTO 70
30 X=RND(1)*30
40 Y=RND(1)*21
50 LOCATEX,Y:PRINT "♥"
60 N=N+1:GOTO 20
70 END
```

(Before executing IF statement)



Whether the program can be executed or not is judged by the IF statement, when it is executed the program branches to the line indicated by the GOTO statement.

(After executing IF statement)



After the program is executed, it is judged by the IF statement. The program jumps conditionally to the GOTO statement.

JUST A MOMENT!

- On multi-statement

A line (not a line on the screen, but one program line) can contain a maximum of 255 characters. More than one BASIC statement may be placed on a line, but each statement must be separated by a colon. The line number 10 composed of many commands i.e multi-statement.

The maximum number of dimensions for array variables is 255 in the "variables and constants", but the numbers of the characters to be assigned are smaller than that because a line contain a maximum of 255 characters.

```
10 COLOR ,1,1:SCREEN
2:FOR R=1 TO 96 STEP
.5:X1=R*COS(R):X2=125
-X1:Y1=R*SIN(R):Y2=10
0-Y1:C=RND(1)*13+2:PS
ET(X2,Y2),C:PSET(X1+1
25,Y1+100),C:PLAY "n=
r:"NEXT R:GOTO 10
```

6. HOW TO RENUMBER PROGRAM LINES (RENUM STATEMENT)

Program is not perfect at the beginning but it becomes complete through being corrected and edited many times.

Since the intervals between lines become to be absurd and look clumsy by inserting and deleting lines, you can correct the old line numbers and renumber the lines by RENUM statement.

Enter as follows:

RENUM ((new number)) ((old number)) ((interval)). () may be omitted.

The defaults for all the contents in the square brackets (()) are 10s.

RENUM also changes all line numbers following GOTO and GOSUB to reflect new line numbers, but if an indistinct line number appears after one of these statements, express it as "Undefined line 000 in XXX" (000 : undefined line number, XXX : the old number)

Be careful about deleting the line.

7. INSERTING EXPLANATORY REMARKS INTO A PROGRAM (REM STATEMENT)

You can hardly recognize which sub program is executed at which line in a

```
10 N=1
20 X=RND(1)*30
30 Y=RND(1)*21
40 LOCATE X,Y:PRINT " "
50 N=N+1 :GOTO 15
60 END
15 IF N>15 GOTO 60
```


long program, (especially when you are watching the program). Therefore, if you insert the foot note called "comment" into the appropriate location the program is easily recognized when it is listed on the screen next time.

The comment is followed by REM statement (non-executable command) or ";" mark (by pressing the keys) that is the abbreviation of REM statement.

(Ex.) Enter "20 REM program" or "20' program."

12-7. FUNCTIONS OF FC-200

We have been mainly explaining the programming through BASIC language up to now.

But in this section we are going to explain how to make the program by way of MSX BASIC which is especially used for FC-200 among several BASIC languages.

You can make pleasant programs because MSX BASIC has many functions of displaying the graphics and playing music, etc.

1. PLAYING MUSIC (PLAY STATEMENT)

There is PLAY command for playing music according to music macro languages. PLAY statement generates the sound with characters called "MUSIC MACRO" command as the following list;

MUSIC MACRO COMMAND		CONTENTS	EXAMPLE
Step of note	A—G	Play the indicated note in the current-octave	Play "CDEFGAB"
	On	Sets the current octave for the following note (n:1—8)	Play "D4CDEFGAB 05 CDE"
	Nn	Plays note n. (n: 1—95) Sets the length of the following note.	Play "N36N37N40"
Length of note	Ln	(n: 1, 1 2, 4, 8, 16, 32, 64) • (period), means its length is multiplied by 3/2	Play "L404C" Play "L204.C"
Rest	Rn	Sets the length of Rest (n: 1, 2, 4, 8, 16, 32, 64) • (period) means its length is multiplied by 3/2	Play "CR4DR4E" Play "CR4DR4.E"
		Sets the number of quarter notes in a	

Tempo	Tn	minute. (n: 32—255)	Play "T250CDE"
Volume	Vn	Sets the volume envelope. (n: 0—15)	Play "V5C V10C V15C"
Tone	Sn	Converts shape of the envelope. (n: 1—15)	Play "S2M300CDE"
	Mn	Converts the period of the envelope	

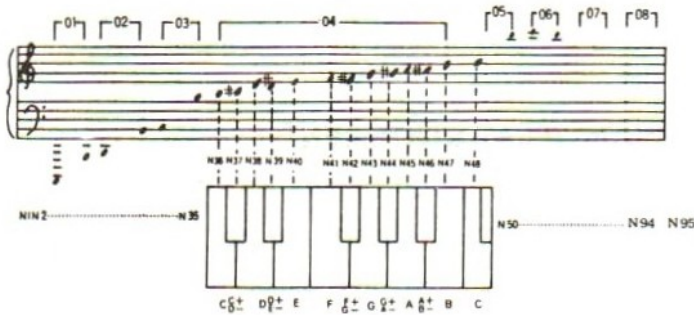
* Refer to SOUND command regarding the envelope and MSX BASIC reference too.

1) • PLAYS NOTE

Do, Re, Mi, Fa, Sol, La, Si, is replaced by C,D,E,F,G,A,B.

Both capital and small letters is available.

The plus sign (+) following English characters represents (sharp), a minus sign (-) does flat (b).



2) LENGTH OF NOTES (Ln)

In setting the length of a note, n has the range from 1 to 64. See the following table;

○ (whole note) = L1	♪ (half note) = L2
♪ (quarter note) = L4	♪ (8th note) = L8
♪ (sixteenth note) = L16	♪ (32nd note) = L32
♪ (sixty-fourth note) = L64	

To express one and a half length of a note, • (period) follows the note.

For example, (quarter note with a period) = L.4 (musical symbol)

More than one period are allowed. In case of a note with 3 periods the length of the note is multiplied by $1.5 \times 1.5 \times 1.5 = 3.375$.

3) NOTE

A note is expressed by musical step and length.

(Ex.) Play "L404G"

4) PAUSE (REST)

Rn represents a rest, and n shows the length of a rest.

▬ (whole rest) = R1

▬ (half rest) = R2

∩ (quarter rest) = R4

∩ (8th rest) = R8

∩ (16th rest) = R16

∩ (32nd rest) = R32

∩ (sixty-fourth rest) = R64

5) TEMPO

Tn stands for the tempo of a note. n means the number of quarter notes played in a minute.

n has the range from 32 to 255. Once you fix the tempo, it will be available until changed again.

6) VOLUME

Vn sets the volume (n has the range from 0 to 15)

7) TONE

Sn sets the shape of an envelope (n has the range from 0 to 15) and Mn sets the period of an envelope. (n may range from 0 to 65535).

S and M are used in pairs, but they are not allowed to be used with V simultaneously.

(Ex.) play "V9CDE"

play "S9M2000CDE"

play "S2M300CDE"

JUST A MOMENT!

THE ORDER OF STATEMENTS

When you play music by using play statement, be careful about the followings;

- Since a sound is generated in the range of the note statement (A—G), you must assign the volume and musical step ahead of the note command.

(Ex.) play "V1504232DC".


- Once the note commands (tempo, musical step, and length of note, etc.) are assigned, they remain until they are assigned again.

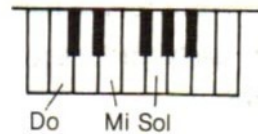
8) CHORD

You can assign the triple chord.

Play (channel A) (Channel B), (Channel C)

character string denoting notes

(Ex.) Play "C" "E" "G"  (Chord of Do, Mi, Sol)



9) VARIABLES

In music MACRO commands, it is allowed to change the note, and tempo by means of variables.

Variables is surrounded by "=" and ";"

"(MUSIC MACRO Command) = (Variable);"

For example, even if you enter `X=8;play N=X` in place of `play "N8"`.

The result will be identical.

(Ex.) The program on the right figure generates all the range of note.

(Ex.) This program generates "Do" note by changing the mood of a note.

```
10 FOR P=1 TO 96
20 PLAY "n=P;"
30 NEXT P
```

```
10 PLAY "t250"
20 FOR S=1 TO 14
30 FOR M=200 TO 3000 STEP 150
40 PLAY "s=s;m=m;c"
50 NEXT M
60 NEXT S
```

2. GENERATING SOUND (SOUND STATEMENT)

By addressing the memory (register) desired among a lot of memories inside the SOUND IC generating the sound and then imputing the data of the sound that you want to generate, it'll be possible for SOUND statement to generate several types of sounds. (ex. wave sound, effective sound, etc.)

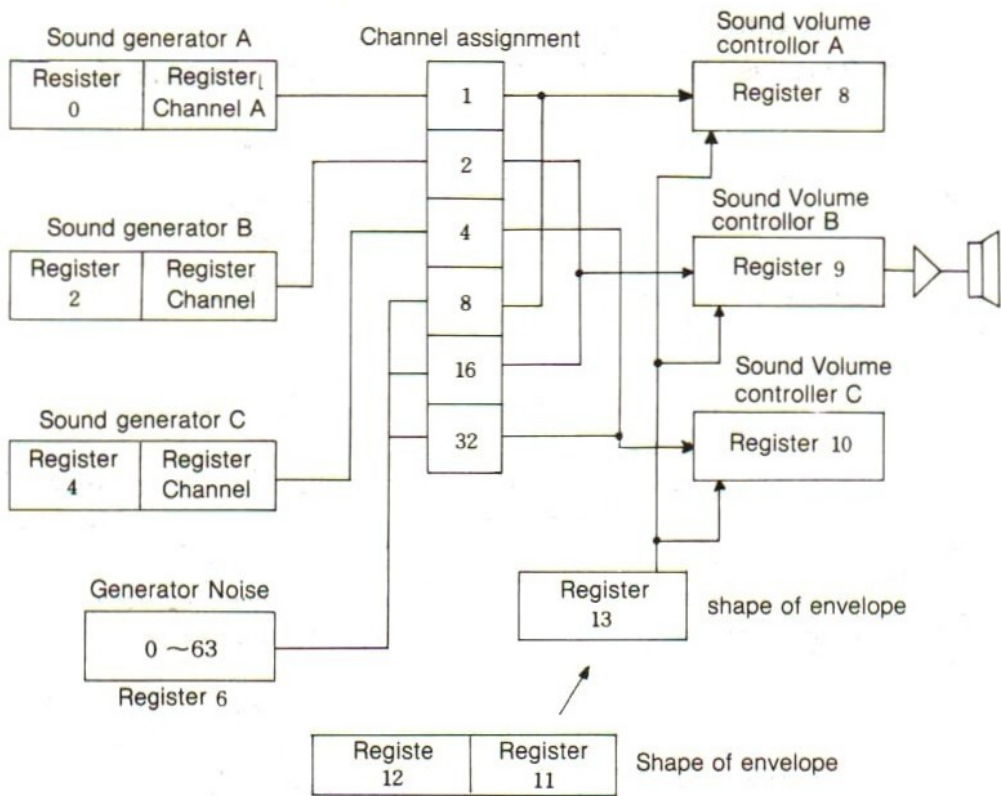
There are 3 control modes in this I.C. as follows;

- Generating the continuous staccatos... The three-sound generators (channel A, channel B, channel C).
- Generating NOISE (Wave sound, etc.)... sharing channel A, B and C.
- Changing the volume automatically (function of envelope)... it is sharing channel A, B, C and Noise.

Sound statement is executed when you enter the following data;

SOUND (Register No) DATA —

Keep in main that DATA can be Integer input or bit input according to Register Number (see the following chart).



SYSTEM CONSTRUCTION OF SOUND IC

1) GENERATING STACCATOS

You need to assign (the channel), (the sound volume) and (the frequency) to generate the staccato.

In the first place, you should assign the channel (the conversions of the channel A, B, C and noise)

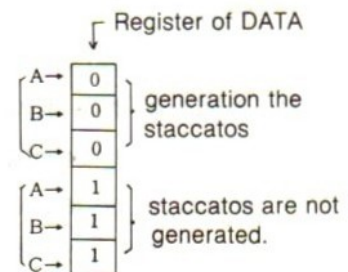
Through the channel assignment, sound is generated if the relevant bit among the input data to Register 7 is 0, on the contrary, it is not generated if it is 1. (Active low)

They are assigned as follows;

SOUND 7, (number of bits)

(Ex.) SOUND 7, & BXX111000

DATA "111000" are input to Register 7 and channel A, generates the sounds of



the channel, A, B, C. (X: See note)

After that, assign the sound volume (Register 8, 9, 10).

The sound volume has the range from 0 level (no sound) to 15 level (maximum) and all the levels are composed of 16 steps.

(Ex.) SOUND 8, 10 

The Registers inside the SOUND I.C. consist of the followings;

REGISTER		DATA							
R0	Channel A frequency	0—255							
R1		0—15							
R2	Channel B frequency	0—255							
R3		0—15							
R4	Channel C frequency	0—255							
R5		0—15							
R6	Noise frequency	0—31							
R7	Channel assignment	IN /OUT		NOISE			CHANNEL		
		IOB	IOA	C	B	A	C	B	A
R8	Volume of channel A	Volume (0—15, 16 with envelope)							
R9	Volume of channel B	Volume (0—15, 16 with envelope)							
R10	Volume of channel C	Volume (0—15, 16 with envelope)							
R11	Period of envelope	0—255							
R12		0—255							
R13	Shape of envelope			E3	E2	E1	E0		
R14	I/O PORT A DATA	8 bit parallel I/O port A							
R15	I/O PORT B DATA	8 bit parallel I/O port B							

* Once the values of Register are assigned, they remain until new data will be assigned to registers.


In the second place, we assign the frequency of the sound.

Channel A is set to Register 0, 1

Channel B is set to Register 2, 3

Channel C is set to Register 4, 5

Refer to MSX BASIC reference about calculating the values set to Registers.

(Ex.) SOUND 0, 12: SOUND 1, 2000 

2) GENERATING NOISE

Noise is generating "Sha—" sound, and is used for making the broken sound. To generate Noise, You should assign "the channel", "the volume of sound", and "the Frequency".

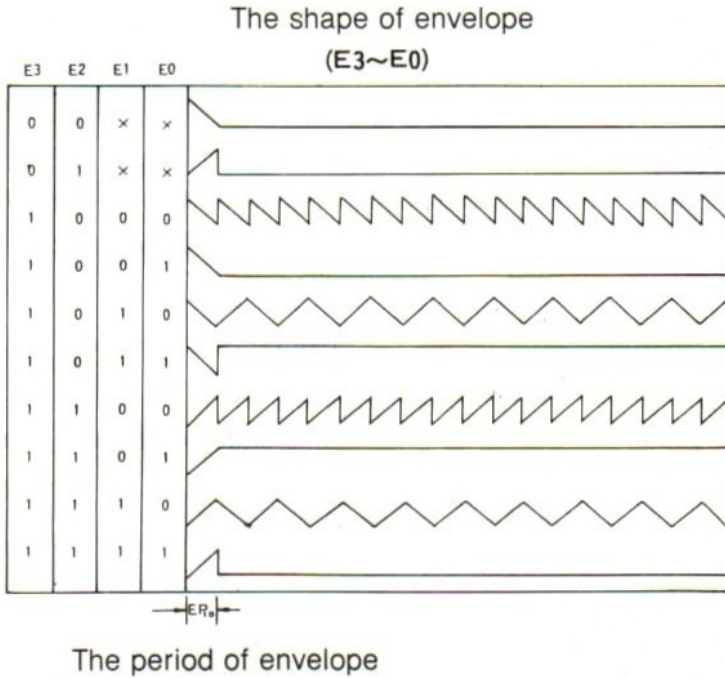
Since the Noise generator is only one, the same Noise from the channel A, B and C is generated.

(Ex.) SOUND 6, 15 

3) GENERATING ENVELOPE

After assigning the channel, we can set the shape of envelope that the volume is changing automatically by setting the value of Register 13. At this time, the shape of envelope is set to register 13 and the period of envelope is set to register 11 and 12.

According to the value of register 13 (E0—E3), the shape of envelope is as follows;



NOTE

“X” mark means that “0” or “1” will be all right.

3. CONTROLLING THE SCREEN (SCREEN STATEMENT)

FC-200 converts a screen into other modes, writes the program, and draws pictures.

SCREEN command assigns the screen mode.

SCREEN MODE is assigned by SCREEN (mode) ((size)),
└── 0—3 ─┘ └── 0—3 ─┘

1) TEXT MODE

TEXT MODE is assigned by entering SCREEN 0 or SCREEN 1 statement. So, the maximum numbers of characters on the full screen are 40 characters (maximum)x24 lines in SCREEN 0 and 32 characters (maximum)x24 lines in SCREEN 1.

The numbers of the character in one line may be changed by the "WIDTH" statement (In the state of power-ON or RESET, 32 character x 24 lines can be displayed in the SCREEN 1 mode).

Even if you assign either SCREEN 0 or SCREEN 1, only the numbers of characters are different but other functions are identical. It does not make any difference that you can assign any one between program mode and direct mode in this text mode. But note that in text mode, all graphic statements except PUT SPRITE statement generate an "illegal function Call" error.

In text mode, you can use the following statements regarding the screen display: Print statement prints the data (characters and symbols) on the screen.

Locate statement determines the position of the cursor.

Cls statement clears the screen.



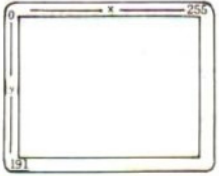
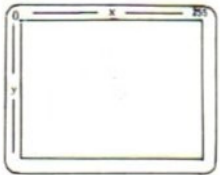
Color defines the colors of characters and the screen (see 4. "Defining the color")

NOTE

If you assign the maximum number of characters by the WIDTH statement, in any circumstances, some characters (of the left and right border) may be cut according to the T.V. set.

2. THE GRAPHIC MODE

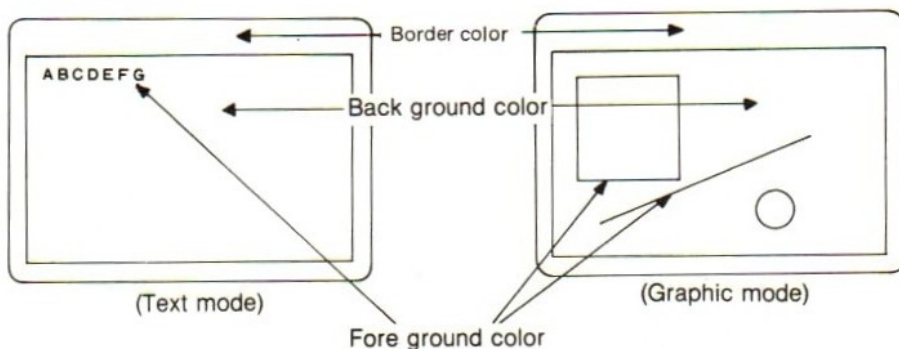
Graphic Mode is set by entering the SCREEN 2 or SCREEN 3 statement. SCREEN 2 is set to select the high resolution mode and SCREEN 3 is set to select the low resolution mode. The program mode is used in the graphic mode. If the program execution is terminated, it automatically returns to the TEXT mode (The mode is forced to return to the Textmode when the 'INPUT' statement is encountered in the process of program execution).

SCREEN MODE		SCREEN statement	Function and screen display
TEXT mode	40x24 Text mode	SCREEN 0	<ul style="list-style-type: none"> Both the Direct Mode and the program mode can be used in this Mode. Graphic statements are hardly used. 
	32x24 TEXT Mode In the state of power-ON the default for the mode is 32x24	SCREEN 1	<ul style="list-style-type: none"> Both the Direct and the mode can be used in this mode. Graphic statements are hardly used. 
Graphic mode	High resolution graphic mode	SCREEN 2	<ul style="list-style-type: none"> Only the program mode can be used. It has 256 dot (horizontal) x 192 (vertical) dot picture elements. 
	Low-resolution graphic	SCREEN 3	<ul style="list-style-type: none"> Only the program mode can be used in this mode It has resolution capacity of 64 blocks (H)x48 blocks (V) (1 Block consists of 4 dots x 4 dots) 

4. DEFINING THE COLOR (COLOR STATEMENT)

We can define 15 colors. Actual colors corresponding to color codes are as follows;

Color Code	Color	Color Code	Color	Color Code	Color	Color Code	Color
0	Transparent	5	Light blue	10	Dark yellow	15	White
1	Black	6	Dark red	11	Light yellow		
2	Medium green	7	Cyan	12	Dark Green		
3	Light green	8	Medium red	13	Magenta		
4	Dark blue	9	Light red	14	Gray		

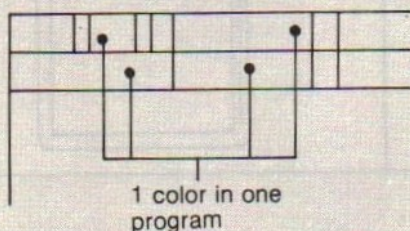


The color statement assigns colors as follows;

COLOR (FORE-GROUND COLOR CODE), (BACK GROUND COLOR),
 CODE
 (BORDER COLOR),
 CODE

NOTE

The fore—ground color is not assigned even by the GRAPHIC statement. But, in high resolution mode (SCREEN 2), keep in mind that you can only assign one fore-ground color within the horizontal 8 bit program. For



example, if you enter a blue point into the coordinate (3.0) and a Red point into (6.0), The color of the point (3.0) turns out to be Red. Like this, the last assigned color is valid in the same program.

5. DRAWING LINES (LINE STATEMENT):

It is used draw a line on the screen in the graphic mode (SCREEN 2, 3)

The LINE statement is entered as follows;

LINE (the start coordinate) — (the end coordinate),

((color code)), (B/BF)

└mode conversion

No sign, draws the line

If 'B' is specified, draw the
rectangle

If 'BF' is specified, fills in the
rectangle

(Ex.) The left program is to draw a
line continuously on the
screen.

```
10 COLOR ,1,1
20 SCREEN 2
30 C1=RND(1)*15
40 C2=RND(1)*15
50 FOR X=0 TO 255 STEP 8
60 LINE(X,0)-(255-X,191),C1
70 LINE(X+4,0)-(251-X,191),C2
80 NEXT X
90 FOR Y=0 TO 191 STEP 8
100 LINE(0,191-Y)-(255,Y),C1
110 LINE(0,187-Y)-(255,Y+4),C2
120 NEXT Y
130 GOTO 30
```

6. SETTING A POINT

The PSET statement is used to set a point on the graphic mode screen. The

PSET statement is entered as follows:

PSET (The coordinate) ((Color))

(Ex.) The left program is to set a
point on the screen.

```
10 SCREEN 2
20 X=RND(1)*255
30 Y=RND(1)*191
40 C=RND(1)*15
50 PSET(X,Y),C
60 GOTO 20
```

NOTE

Within one block (4x4 dots) in the mode of low resolution.

All coordinates are identical

(Ex.) PSET (1, 1) = PSET (3, 3)

7. DRAWING A CIRCLE (CIRCLE STATEMENT)

CIRCLE statement is as follows;

CIRCLE (center coordinate), radius, color, ((start angle, end angle, aspect ratio)). If the contents in the square bracket (()) are omitted, the defaults for the start angle, the end angle and the aspect ratio will be 0° , 360° and 1 respectively.

A radian is used as a unit of angles.

```
10 COLOR 15,1,1
20 SCREEN 2:C=2
30 FOR X=0 TO 255 STEP 16
40 Y1=SIN(4*3.1416*X/255)*30+50
50 Y2=COS(4*3.1416*X/255)*30+140
60 CIRCLE(X,Y1),7,C
70 CIRCLE(X,Y2),7,C+1
80 C=C+1:IF C>14 THEN C=2
90 NEXT X
100 GOTO 100
```

(Ex.) $360^\circ = 2 \text{ radians} = 2 * 3.1416$
Radians

(Ex.) The right program is to draw a circle continuously by means of sine and cosine functions.

8. PAINTING (PAINT STATEMENT)

The PAINT statement is used to fill in an arbitrary graphic figure drawn by the LINE statement and CIRCLE statement with one color.

PAINT (start coordinate), (specified color), (border color) fills in a drawn graphic figure with the specified color.

(Ex.) If you insert the right program between the appropriate lines of the previous program that draws a circle, you can fill in the circle with the specified color

```
65 PAINT(X,Y1),C
75 PAINT(X,Y2),C+1
```

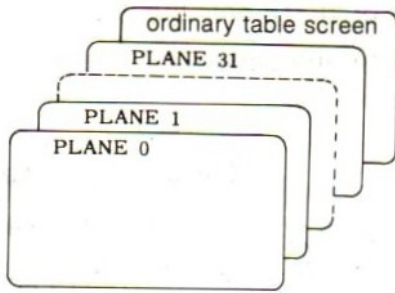
9. ACTIVATING PICTURES (SPRITE FUNCTION)

By repeating the PSET statement and changing coordinates you can activate the pictures on the screen. But much more convenient way called "SPRITE function" is available in MSX BASIC.

1) WHAT IS SPRITE?

When you activate one picture, it is inconvenient to move one point by repeating the PSET and PRESET statement.

The SPRITE function is to consider a picture as a pattern and activate it as a pattern.



THE SPRITE functions are executed by the SPRITE \$ () statement that assigns the SPRITE pattern and as PUT SPRITE statement that sets up the sprite attributes.

There are the virtual screen that has the range from 0 to 31 in MSX BASIC and we call it "PLANE".

PLANE is assigned regardless of the types of MODE (TEXT, GRAPHIC, DIRECT PROGRAM).

It is allowed to assign one (sprite pattern on one sheet of plane. And the sprite size (sprite pattern size) is determined by the SCREEN statement. You must assign the SCREEN, before executing the SPRITE \$, PUT SPRITE statements, as follows;

SCREEN (SCREEN MODE), (SPRITE SIZE)

(Ex.) SCREEN 2, 2

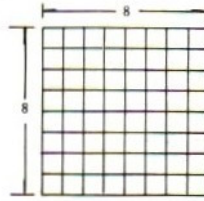
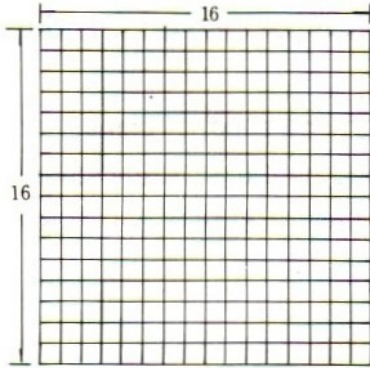
→ The number that determines the size of SPRITES.
→ MODE number

Number	Sprite Size
0	8x8 unmagnified
1	8x8 magnified
2	16x16 unmagnified
3	16x16 magnified

explanation of terms
 SPRITE: nymph
 PLANE: flat panel

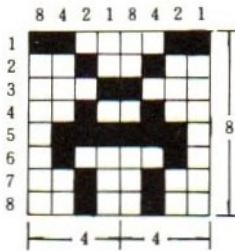
2) HOW TO ASSIGN THE SPRITE PATTERN (SPRITE \$ () STATEMENT)

SPRITE patterns are set to 8x8 dots or 16x16 dots but you need to assign some values (0 or 1) to each dot to set the pattern.



If you want to display a dot, you have to assign 1 to the dot (bit). If you don't, you have to assign 0 to the dot. There are following 2 methods to convert dots gathering into bit values;

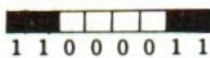
- Enter the data into the program directly.
- Enter the data by converting the values into the binary number, the octal number, the decimal number, or the hexadecimal number.



Let's take the left figure for an example. To describe the pattern on line 1.

DATA must be entered as follows;11000011.

If you save the data to the computer, it will generate the following pattern;



■ ■s are displayed.

In order to set the pattern you have to use the SPRITE\$ () statement.

The above—drawn pattern is set as follows;

```
SPRITE $ (n) = CHR $ (& B11000011) + CHR $ (& B00100100) + CHR $
(& B00011000) + CHR $ (& B00100100) + CHR $ (& B01111110) + CHR
$ (& B01000010) + CHR $ (& B00100100) + CHR $ (& B00100100) □
```

(n is the pattern number of sprite)

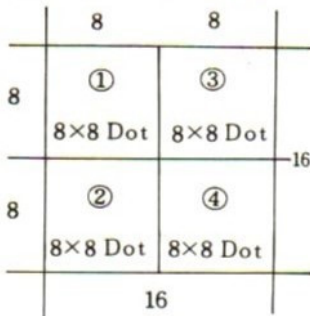
The above-method uses binary numbers. (& B...) can be replaced by the octal number (& O...) or the hexadecimal number (& H...).

Explanation of Terms Dot and Bit

1 A dot means one glaring point displayed on the screen, and characters and symbols are composed of many dots. A bit is a unit that is used transmit the instruction to the computer. A bit has the value of 0 or 1, and one instruction or information are composed of several bits.

In case of 16x16 SPRITE pattern, you'd better assign four 8x8 sprite patterns in the following order;

SPRITE \$(n) = (\text{assign (1)}) + (\text{assign (2)}) + (\text{assign (3)}) + (\text{assign (4)})



(see the left figure)

We call SPRITE \$(n)\$ "pattern number n". n has the range from 0 to 255 in the 8x8 sprite pattern, and from 0 to 63 in the 16x16 sprite pattern.

(Ex.) SPRITE \$(11) = \text{"_"}\$
 pattern number data of pattern

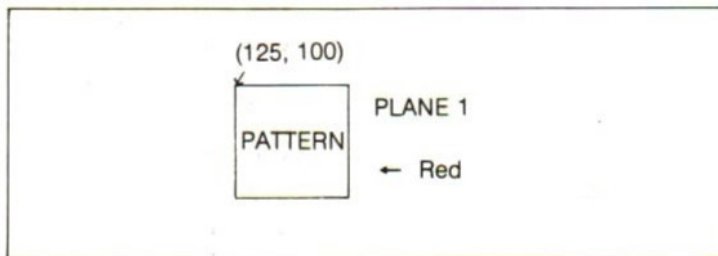
3) GENERATING SPRITE (PUT SPRITE STATEMENT)

If you write down the assigned pattern on the plane, you have to use the PUT SPRITE statement as follows;

PUT SPRITE (Sprite plane number), (coordinate), (Color code), (pattern number).

Let's take an example as follow;

If you enter PUT SPRITE 1 (125,100), 8,1 \square , SPRITE PATTERN 1 of the coordinate (125,100) on the PLANE 1 is assigned to RED (color code 8).

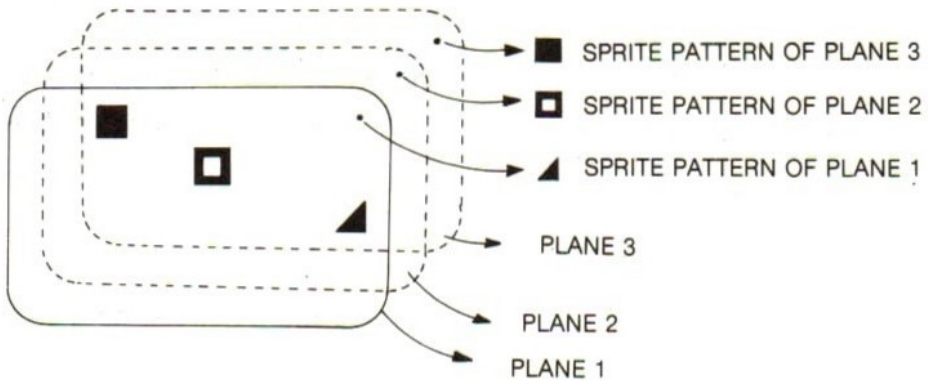


Only SPRITE PATTERN should be assigned to a plane.

If you assign sprite patterns by using several planes, the fifth pattern will not be displayed when the patterns are overlapped in the horizontal direction (when the y coordinates are almost identical).

And there are priorities in displaying planes, planes are displayed from those of small numbers in order when they are overlapped.

The following program shows that SPRITE PATTERN 1, 2 and 3 are assigned and displayed on PLANE 1, 2, 3 respectively after that.

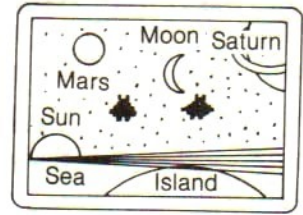


```
5 COLOR 15,1,1:SCREEN 2,1
10 FOR S=1 TO 3
20 S$=""
30 FOR T=1 TO 8
40 READ A$
50 S$=S$+CHR$(VAL("&b"+A$))
60 NEXT T
70 SPRITE$(S)=S$
75 PUT SPRITE S,(S*70,S*50),S*2,S
80 NEXT S
90 GOTO 90
100 DATA 11111111
110 DATA 11111111
120 DATA 11111111
130 DATA 11111111
140 DATA 11111111
150 DATA 11111111
160 DATA 11111111
170 DATA 11111111
200 DATA 11111111
210 DATA 11111111
220 DATA 11000011
230 DATA 11000011
240 DATA 11000011
250 DATA 11000011
260 DATA 11111111
270 DATA 11111111
300 DATA 00000001
310 DATA 00000011
320 DATA 00000111
330 DATA 00001111
340 DATA 00011111
350 DATA 00111111
360 DATA 01111111
370 DATA 11111111
```


13. HOW TO DRAW THE ANIMATION

— Practising programming —

We make the simple moving figure and generate sound in this chapter. We have been explaining basic statements to operate the FC-200 so far. In this chapter, we draw figures like that in the right write the program, and explain the following statements; It might be a little more difficult for you to understand it, but you will be able to draw more complicated figures when you get used to them.



(Summary of the program)
The programs draw the figure like the above one. And it generates the sound by the play statement. UFO is activated by the PUT SPRITE statement.

Statement and function available

SCREEN	COLOR	GOTO
RND	LINE	FOR-NEXT
CIRCLE	PAINT	SWAP
OPEN	PSET	MOD
PRINT	SPRITE\$	SIN
PLAY	PUT SPRITE	CLOSE

13-1. DRAWING THE STARS IN THE SKY (SCREEN, COLOR, RND)

We set the color of the sky and draw 100 stars. You assign auto mode by entering

AUTO 50

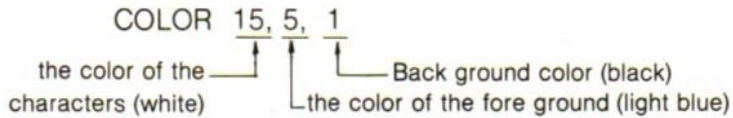
```

auto 50 ← entering the program from line 50
50 COLOR 15,5,1 ← set the color of the screen
60 SCREEN 2,3 ← assign SCREEN 2, magnify the SPRITE to 16x16
70 FOR N=1 TO 100 ←
80 X=RND(1)*255:Y=RND(1)*191 ← the positions of the
90 PSET(X,Y),11 ← stars are random,
100 NEXT N ← colors are yellow draw 100 stars.
110
800 goto 800 ← go to infinite loop in the state of SCREEN 2.
    
```

You can enter command clause (REM statement) until line 50.

The color of the screen is determined by the COLOR statement on line 50.

The COLOR statement is as follows;



You can change the colors as you want, but use the above colors for the time being.

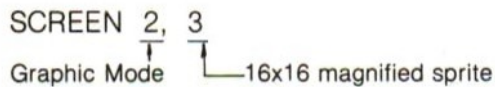
Explanation of terms

INFINITE LOOP

GOTO 800 clause on line 800 is the statement, "branches out of line 800 to line 800." Line 800 entered to show the current state of the SCREEN. The computer is shown just as it is stopped in the state of SCREEN 2 by this statement.

If you want to hold the program, press the **CTRL** + **STOP** keys.

You can assign the Graphic Mode by the SCREEN statement on line 60, and set up the 16x16 magnified sprite for UFO that will be entered later. The 8x8 magnified sprite will be all right, but the 8x8 unmagnified (or 16x16 unmagnified) sprite will make UFO a little smaller.



Next, we draw the stars. The programs is as follows;

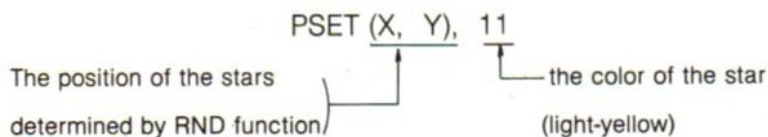
X = RND(1)*255:Y = RND(1)*190

PSET (X, Y), 11

The positions of the stars are determined by the RND function.

The RND function assigns the approximate X, Y coordinates (it generates a random number between 0 and 1).

255 and 190 are the magnitudes of the horizontal side and the vertical side respectively. The magnitudes of both sides are multiplied by 255 and 190 since the results of RND function only are limited to the numbers between 0 and 1 (0, 0.1..., 0.2..., 1). Once the positions of the stars are fixed, the stars are drawn by the PSET statement.

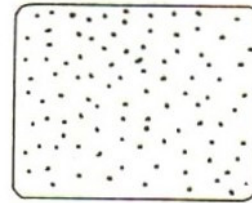


The numbers of the stars are determined by the FOR—NEXT statement from lines 70 through 100.

```
70 FOR N= 1 TO 100
100 NEXT N
```

the process is repeated 100 times.

If you enter the program up to here, the right figure will be displayed. Enter RUN again (terminate AUTO mode by the **CTRL** + **STOP** keys and press the **F5** key). When you hold the program, type the **CTRL** + **STOP** keys. If the program is not executed normally, see if the program is all right by entering LIST.



13.2. DRAWING LINES

Let's draw the wave lines on the sea, and 8 solar rays. Each position is assigned as follows; (Type AUTO 110 key's and enter the program from line 110)

Explanation of terms

RND: Refer to MSX BASIC reference about RND function.

PSET: The above-program is composed of 2 lines for better understanding.

PSET (RND (1) * 255, RND (1) * 190), 11 will be O.K.

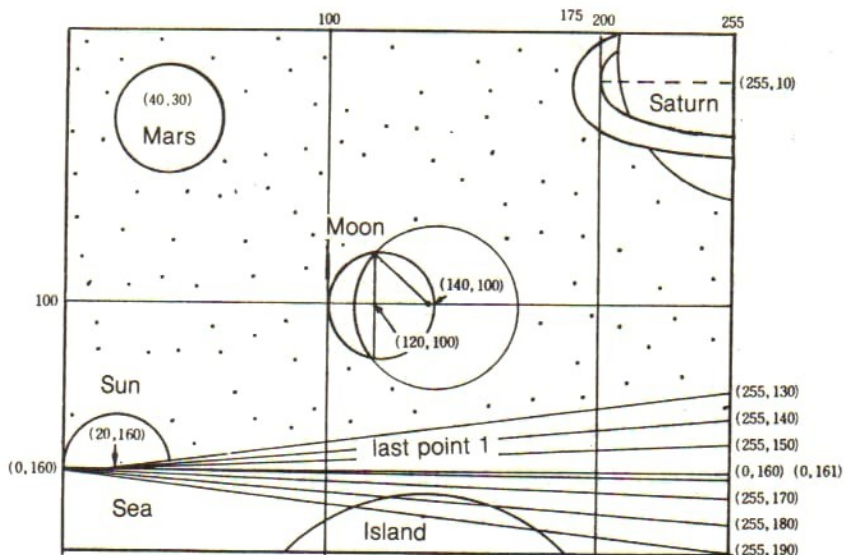
```
auto 110
110 LINE(0,160)-(255,160),15 ← draw the horizontal line in white.
120 LINE(0,161)-(255,161),15 ←
130 LINE(0,162)-(255,191),4,BF ← draw the sea as a blue box.
140 LINE(20,160)-(255,130),8 ←
150 LINE(20,160)-(255,140),8 ← draw the solar rays in red.
160 LINE(0,161)-(255,150),8 ←
170 LINE(0,162)-(255,170),15 ←
180 LINE(0,162)-(255,180),15 ← draw the sea in white.
190 LINE(0,162)-(255,190),15 ←
200
```

LINE (X1, Y1) — (X2, Y2), C (,BF)

- | | | | |
|------------------|----------------|------------|---|
| Start coordinate | end coordinate | line color | <ul style="list-style-type: none"> • If no mark, draw the line • If B is specified, draws a rectangle • If BF is specified, paints the rectangle |
| | | | |
| | | | |

This line statement causes the horizontal line to be drawn in white with lines 110 and 120, and the sea to be drawn blue by line 130. Once the program is entered, terminate AUTO mode and enter RUN just like before.

When the lines are drawn, go over the next chapter.



Explanation of terms LINE

The start point of the line statement can be omitted. In this case, the last values of X,Y (The values used for PAINT, LINE and CIRCLE) are used as the last point

(Ex.) LINE — (X, Y), C

13.3. DRAWING THE STARS AND THE MOON AS CIRCLES (CIRCLE, PAINT STATEMENTS)

* Drawing Saturn, Mars, the Sun and the Moon.

You can draw circles by the CIRCLE statement. The CIRCLE statement is used as follows;

(Refer to MSX BASIC reference for details regarding the CIRCLE statement)

CIRCLE (X, Y), L, C (, R1, R2, S)

The center coordinates of the circle, radius, color, start angle (radian)

end angle (radian)

aspect ratio (the ratio of horizontal

to vertical of a circle)

- If S is 1 it is right circle
- If S is less than 1, it is an ellipse with the long horizontal axis.
- If S is more than 1, it is an ellipse with the long vertical axis.

Type Auto 200 and then input the program from line 200.

```

auto 200
200 'circle ← command clause; explains the following program;
210 R=3.141593#/180 ← the value of variable R is determined
220 CIRCLE(255,10),50,10
230 PAINT(255,10),10,10 } ← draws the big circle saturn
240 CIRCLE(255,10),70,14,R*130,,.25
250 CIRCLE(255,10),80,14,R*100,,.35 } ← draws the ring of saturn
260 PAINT(255,35),14,14
270 CIRCLE(40,30),20,9 } ← draws Mars
280 PAINT(40,30),9,9
290 CIRCLE(120,100),20,11,R*100,R*265,1 } ← draws the Moon
300 CIRCLE(140,100),30,11,R*140,R*225,1 }
310 PAINT(105,100),11,11
320 CIRCLE(20,160),20,8,0,R*180,1 } ← draws the Sun
330 PAINT(20,150),8,8
340 CIRCLE(155,370),200,3,R*50,R*130,1 } ← draws the Island
350 PAINT(155,185),3,3
360

```

- How to enter command clause -variables, etc.

```

200 'circle
210 R=3.141593#/180

```

The comment clause on line 200 is used as the remark saying that "a circle is drawn from here;" when you read the program again. The value of R = 3.141593 # /180 on line 210 is converted into the angle unit for better understanding when you correspond the numeric value to the Radian argument that is used on line 240.*

Explanation of terms	π (PI)						
Radian is corresponded to angle unit as follows;							
Radian	0	$\pi/6$	$\pi/4$	$\pi/2$	π	$3\pi/2$	2π
Angle	0°	30°	45°	90°	180°	270°	360°

$\pi = 3.141593$

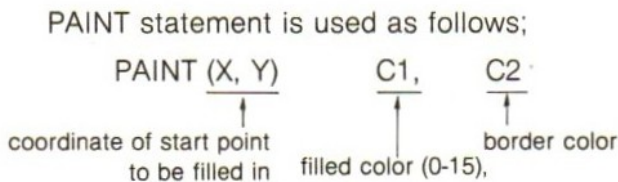
- Drawing Saturn

The central coordinates of Saturn is set to (255, 10) and the color is set to the dark yellow (color code; 10). And fill in the circle in yellow.

```

220 CIRCLE(255,10),50,10
230 PAINT(255,10),10,10

```



Next, the ring of saturn is drawn as follows;

```

240 CIRCLE(255,10),70,14,R*130,,.25
250 CIRCLE(255,10),80,14,R*100,,.35
260 PAINT(255,35),14,14

```

Statement 240 draws the inner ellipse, the outer ellipse is drawn by statement 250, and statement 260 fills in the ring in color.

Let's take line 240 for instance.

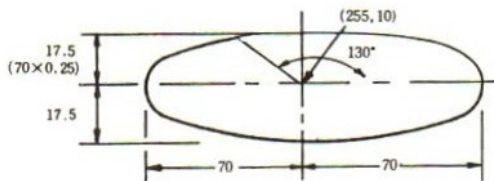
```
CIRCLE (255, 10), 70, 14, R * 130,,.25
```

The center of the ellipse is determined by (255, 10).

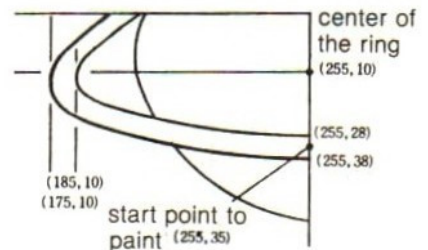
This location shows about a-quarter of the ellipse. 70 means the radius and 14 means the color of the ellipse, gray (color code: 14).

"R*130,,.25" determines of the ellipse.

Finally, the ellipse is drawn from 130° point to another 360° point by statement 240.

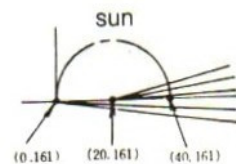


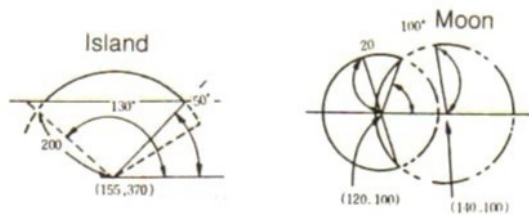
The horizontal radius is 70, and the vertical radius is 17.5 (70x0.25). The position of PAINT on line 260 is set between the inner ellipse and the outer one like the right figure. Test and see what could happen, if the position is set wrong.



- Drawing Mars, the sun, the moon and the island.

Mars is located on (40, 30) with the magnitude of 20. The sun, the moon and the island are drawn like the following figures. Since painting by the PAINT statement comes through the aperture between lines, be careful when you enter the values.





When you finish entering the program up to line 350, you terminate AUTO mode and enter the RUN statement.

13-4. ENTERING CHARACTERS

Enter the follow programs from line 400 to display the letters.

```

auto 400
400 'print
410 OPEN "GRP:" FOR OUTPUT AS#1
420 PSET(220,10):PRINT #1, Saturn
430 PSET(30,30),9:PRINT #1, Mars
440 PSET(130,80),4:PRINT #1, Moon
450 PSET(170,180),3:PRINT #1, Island
460 PSET(25,170),4:PRINT #1, Sea
470 PSET(10,130),4:PRINT #1, Sun
480 CLOSE
490

```

The symbols and the characters stored in the computer can be displayed on the graphic screen by the OPEN statement. The line 410 opens the computer files for displaying something on the screen. # 1 is the file name. The file number is assigned to # 1 when symbols and characters are displayed on the graphic screen. The characters following PRINT # (file name) will be displayed on the screen, but the PSET statement sets the display coordinates in order to determine the display position of the characters and symbols. When you have finished displaying the characters, you close the computer file by the CLOSE statement. (Refer to MSX BASIC reference for details)

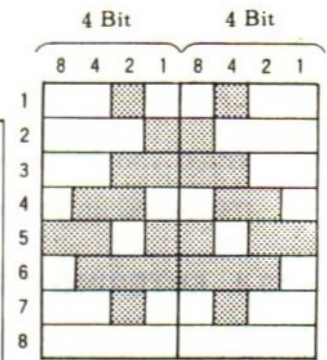
13-5. ACTIVATING U.F.O. (SPRIFE \$, PUT SPRITE)

The sprite is a pattern which one picture element is composed of 8x8 dots. One figure is activated freely by using the sprite. The number of SPRITE pattern may be assigned up to 256 when the size of the sprite is 8x8 dots (SCREEN 2, 0 or SCREEN 2, 3), and may be assigned up to 64 when the size of the sprite is 16x16 dots (SCREEN 2, 1 or SCREEN 2, 3). But the length of the simultaneous display, that is, pattern number is fixed to 32 and each has its own priority to be displayed.

```

500 'sprite
510 A1$=CHR$(&H24) ← Assigns the pattern of the 1st row from above.
520 A2$=CHR$(&H18) ← Assigns the pattern of the 2nd row from above.
530 A3$=CHR$(&H3C) ← Assigns the pattern of the 3rd row from above.
540 A4$=CHR$(&H66) ← Assigns the pattern of the 4th row from above.
550 A5$=CHR$(&HDB) ← Assigns the pattern of the 5th row from above.
560 A6$=CHR$(&H7E) ← Assigns the pattern of the 6th row from above.
570 A7$=CHR$(&H24) ← Assigns the pattern of the 7th row from above.
580 A8$=CHR$(&H0) ← Assigns the pattern of the 8th row from above.
590 A$=A1$+A2$+A3$+A4$+A5$+A6$+A7$+A8$
600 SPRITE$(5)=A$
610 X1=-15:X2=255:K=2
620 FOR X=X1 TO X2 STEP K ←
630 Y=SIN(T)*30+50+X*.25 ← assigns the movement of sprite 1
640 Y1=X*.8-30 ← assigns the movement of sprite 2
650 PUT SPRITE 5,(X,Y),15,5
660 PUT SPRITE 6,(X*1.5,Y1),11,5
670 T=T+R*18:NEXT X ←
680 SWAP X1,X2:K=-K ← reverses the values of x1 and x2
690 GOTO 620 ← UFO moves reversely

```



assigns the pattern of sprite 1

Varies x from x1 to x2

Explanation of terms SWAP statement

It is used to exchange the values of two variables

$$X3 = X1$$

If this statement is not used, one more

$$X1 = X2$$

Variables will be needed like in the right equations

$$X2 = X3$$

JUST A MOMENT!

• HOW TO ASSIGN THE PATTERN

A pattern is composed of 8x8 dots, and the assignment of the pattern is made by entering the binary data, the octal data, the decimal or hexadecimal data. (The following programs have the same pattern).

• When the binary data are entered

Convert the pattern into binary value (the combination of 0 and 1). These are entered as
 $A\$ = \text{CHR} \$ (\& B \dots)$

means binary number

enter 0 and 1 according to the screen display

```

500 'sprite
510 A1$=CHR$(&B00100100)
520 A2$=CHR$(&B00011000)
530 A3$=CHR$(&B00111100)
540 A4$=CHR$(&B01100110)
550 A5$=CHR$(&B11011011)
560 A6$=CHR$(&B01111110)
570 A7$=CHR$(&B00100100)
580 A8$=CHR$(&B0)

```

(input of binary data)

```

500 'sprite
510 A1$=CHR$(&H24)
520 A2$=CHR$(&H18)
530 A3$=CHR$(&H3C)
540 A4$=CHR$(&H66)
550 A5$=CHR$(&HDB)
560 A6$=CHR$(&H7E)
570 A7$=CHR$(&H24)
580 A8$=CHR$(&H0)

```

(input of hexadecimal data)

It is very simple since the value of every bit is corresponded to each dot respectively.

- When the hexadecimal data are entered Express the pattern as the hexadecimal number.

They are entered as follows;

A\$ = CHR \$ (& H ...)

↑ means hexadecimal number.

Though the program is a little shorter, but the calculations of the values assigned are very complicated.

- When DATA statement is entered DATA statement assigns the pattern in the program. Assigning and converting patterns is easy, but the program becomes long.

- Assigning the sprite

The pattern is converted into one string line of A\$ on line 590. The U.F.O. pattern of A\$ is assigned to the 5th sprite pattern sprite on line 600.

Lines 650 and 660 assign the 5th sprite pattern to the plane 5 and 6.

In order to activate trapping of the sprite the sprite patterns should be set and the pattern number assigned to each pattern as mentioned above.

And after that, "which sprite pattern will be assigned to which plane and where?" must be defined.

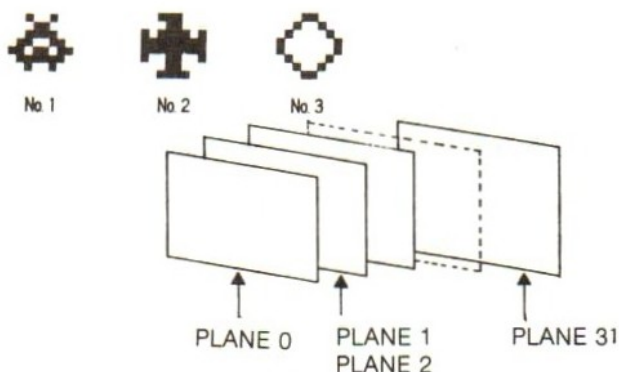
In case of the magnified sprite, the pattern number must be less than 64.

In case of the unmagnified sprite, must be less than 256.

```

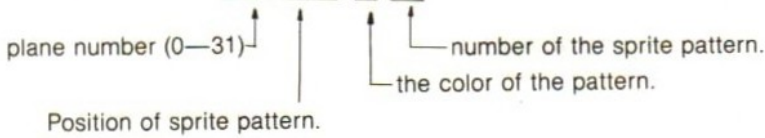
500 'sprite
510 B$=""
520 FOR N=1 TO 8
530 READ B$
540 A$=A$+CHR$(VAL("&B"+B$))
550 NEXT N
560 DATA 00100111
565 DATA 00011000
570 DATA 00111100
575 DATA 01100110
580 DATA 11011011
585 DATA 01111110
590 DATA 00100100
595 DATA 00000000
    
```

(input of DATA)



SPRITE \$ (n) = (string line)

PUT SPRITE (n) (X, Y), C, m



NOTE

- The assignment of sprite positions is done on the upper-left corner of 8x8 dot and 16x16 dot patterns.
- The program from line 510 to line 600 is identical to the followings:
600 SPRITE \$ (5) = CHR\$ (& H24) + CHR \$ (&18)
+ CHR \$ (& H30) + CHR \$ (& H7E)
+ CHR \$ (& H24) + CHR \$ (& H0) Position to be assigned.

- Activating trapping of sprite

You can activate trapping of SPRITE with the infinite loop.

```
600 SPRITE$(5)=A$
610 X1=-15:X2=255:K=2 ← assigns the initial value and step
620 FOR X=X1 TO X2 STEP K ←
630 Y=SIN(T)*30+50+X*.25 ← calculates the movement of plane 5
640 Y1=X*.8-30 ← calculates the movement of plane 6
650 PUT SPRITE 5,(X,Y),15,5 ← sets up sprite attributes on plane
660 PUT SPRITE 6,(X*1.5,Y1),11,5 ← re-assigns the initial value and step
670 T=T+R*18:NEXT X ←
680 SWAP X1,X2:K=-K ← and get the U.F.O to be moved
690 GOTO 620 ← in the reverse direction.
```

Loop

The explanation of the above program.

The 5th pattern of sprite is assigned on line 600. The moving range, the start position, and the end position of the U.F.O are assigned to have the U.F.O moved continuously on line 610. Lines 620 and 670 are FOR—NEXT clause that have the 5th pattern of sprite moved from X1 position to X2 position by incrementing the value of step K.

Line 630 calculates the movement of PLANE 5 (in the vertical direction). Watch out how it is moving. The movement of PLANE 6 is calculated on line 640 (in the vertical direction).

The positions where sprite attributes are set up on PLANE 5, 6 are assigned on line 650 and line 660.

Line 670 determines the range that the values of Y1 are varying.

Line 680 reverses the values of the start point and the end point of X1 and moves the sprite in the reverse direction by varying the value of step. See if it is moving smoothly.

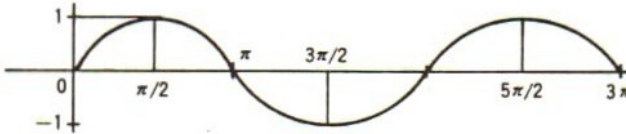
You can see that the two U.F.Os are flying in the space.

If it does not move very well, check the program again.

Explanation of terms SIN FUNCTION

This is one of MSX BASIC FUNCTIONS. In $\text{SIN}(X)$, X is radians.

$\text{SIN}(X)$ has a value between -1 and 1 .



13-6. GENERATING THE SOUND (PLAY STATEMENT)

Enter the right program. While X entered earlier on line 620 is varying from $X1$ to $X2$,

```
645 IF (X MOD 5)=0 THEN PLAY  
"U1506L64DC"
```

this program generates the sound when X is divided by 5 with no remainder by Modulus Arithmetic.

Modulus arithmetic yields the integral remainder of the division.

You can generate the sound you like by changing the sound of PLAY statement, and by entering the repertoire (the character string) make a pleasant program.

JUST A MOMENT!

Blinking the window of the U.F.O

You can blink the window of the U.F.O. flying down from the top-left corner.

Enter the following program;

The window of the U.F.O is assigned to SPRITE 6 on line 615.

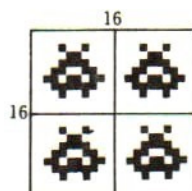
Blinking the window is executed by setting SPRITE 6 to plane 7 in white or transparent through the multi-statements on line 665.

Line 675 have the same function as line 665. Since there is no need to blink the U.F.O. when it flew up and disappeared, IF K = 0, clause determines the conditions.

```
615 SPRITE$(6)=CHR$(0)+CHR$(0)+CHR$(0)+C
HR$(%H18)+CHR$(%H24)
665 FOR J=0 TO 2:PUT SPRITE 7,(X*1.5,Y1)
,15,6:PUT SPRITE 7,(X*1.5,Y1),0,6:NEXT J
675 IF K>0 THEN FOR J=0 TO 4:PUT SPRITE
7,(X*1.5,Y1),15,6:FOR S=0 TO 100:NEXT S:
PUT SPRITE 7,(X*1.5,Y1),0,6:NEXT J
```

Sprite pattern that was mentioned up to now was assigned to 16x16 magnified type (SCREEN 2, 3), but it only uses a-quarter of entire SPRITE pattern screen. (see the right figure.)

In assigning the pattern on the full size of 16x16, 4 patterns may be inserted if you write line 600 as follows;
600 SPRITE \$(5) = A\$ + A\$ + A\$ + A\$



14. MAINTENANCE AND AFTER-SERVICES

14-1. MAINTENANCE

- **With a soft cloth**

- Rub with a soft cloth.

When FC-200 body surface is dirty and covered with the dust, rub the FC-200 body with a soft cloth. If you use a rough one. The surface of the body will be scared.

- In case of heavy dirt

After wetting a soft cloth with the neutral detergents and water, and squeezing it, clean and rub the body with it. Never use chemicals such as thinner, benzene, etc.

14-2. WARRANTY AND AFTER-SERVICE

- Be sure to get the warranty.

The warranty is enclosed with the product.

Confirm are the articles with the agency where you purchased this product warranted and keep it well.

- Warranty is valid for 1 year from the day you purchased the product.

And since it may cost a little to fix it in any circumstance even during warranty, period pay attention to the warranty.

- When the computer is out of order.

Take a careful look at before asking for AFTER SERVICE in the next page; If you can't repair it according to the next page, consult with the agency where you purchased the product.

- Repairs after the warranty period.

About the repair after the warranty period, consult with the agency where you just purchased the product.

If the system is repairable. We will always repair it at your cost in response to you request.

14.3. BEFORE ASKING FOR SERVICE

If GoldStar does not work normally, carry out the following checks before requesting service.

Symptom	Check list
No screen display	<ol style="list-style-type: none">1. Power indicator on keyboard not on—check cord is correctly plugged into the wall socket and that the socket is switched on.2. Check that the T.V. is on3. Check that cable from the RF modulator to the T.V. is connected.4. A ROM Cartridge has been inserted, check that it is the right way round.5. Check T.V. Channel tuning and brightness and contrast setting are correct.
Poor quality or instable T.V. display	<ol style="list-style-type: none">1. Adjust T.V. fine tuner for the appropriate channel selection.2. Adjust the vertical and horizontal hold of T.V. (refer to T.V. operating manual for details)2. Check that the video connecting cable is securely connected to T.V. and computer.
Failure to operate after ROM cartridge insertion	<ol style="list-style-type: none">1. Check that the ROM cartridge is inserted fully.2. Turn the power switch off and on again, waiting at least 5 seconds in between.
Faulty operation of computer	<ol style="list-style-type: none">1. It is possible that interference from the other electrical devices is causing the computer to malfunction. The use of a mains plug incorporating.2. Check that any peripheral devices (printer, cassette tape recorder etc.) are connected securely.3. Check errors are not being generated by the program. It is not unusual for faults generated by a program to be wrongly attributed to machine faults.
Faulty loading or saving to cassette recorder	<ol style="list-style-type: none">1. Check that the cable in the recorder is securely connected to the recorder and to the computer.2. Check the LOAD/SAVE instructions carefully.3. Where possible always operate the cassette recorder from the mains, but if using the batteries make sure that the batteries are not run down.4. Try loading at a number of different volume level settings to see if a good LOAD can be achieved.

15. EXPLANATION OF COMMANDS.

Standard Statement (Concluding the statement regarding files)

LIST/LLIST

format: LIST ((start line number)) ((end line number))
LLIST ((start line number)) ((end line number))
function: To list all or part of the program on the CRT/printer.

NEW

format: N E W
function: To delete the entire program from the memory and reset all variables.

RENUM

format: RENUM ((new line number)) (, (old line number)) (, (INCREMENT))
function: To renumber program lines.

SAVE/LOAD/MERGE

format: 1 SAVE'' (Device-descriptor)
2 LOAD'' (Device-descriptor)
3 MERGE'' (Device descriptor)
function: 1 To save a program file to the device
2 To load a program file from the device
3 To merge lines into the program currently in memory. In this case, all data should be ASCII mode.

RUN

format: RUN ((line number))
function: To execute a program

BLOAD/BSAVE

format: BSAVE (Device descriptor), (top address), (end address), (, (execution address.))
function: To save data of machine language to the indicated device. If (execution address) is omitted, (top address) is regarded as (execution address)
format: BLOAD (Device descriptor) (, R) (, (offset)).
function: To load data of machine language from the specified device. If R op-

tion is specified, after loading, the program begins execution automatically from the address which is specified at BSAVE.

CSAVE/CLOAD

format: CSAVE (file name) (, S)
function: If S option is omitted, BASIC program is saved to the cassette tape. If S option is specified, the data on the screen is saved.
format: CLOAD 'file name'
function: To load a BASIC program file and the data on the screen from the CMT.

TRON/TROFF

format: TRON/TROFF
function: To trace the execution of the program statement.

CLEAR

format: CLEAR ((string space)) (, (highest location))
function: To set all numeric variables to zero, all string variables to null, and close all open files, and optionally, to set the end of memory.

DATA

format: DATA (list of constants)
function: To store the numeric and string constants that are accessed by the program's READ statement(s).

DIM

format: DIM (list of subscribed variables)
function: To specify the maximum variables for array variable subscripts and allocate storage accordingly.

DEFINT/SNG/DBL/STR

format: DEFINT (ranges of letters)
DEF SNG (ranges of letters)
DEF DBL (ranges of letters)
DEF STR (ranges of letters)
function: to declare variable type as integer, single precision, double precision, or string.

DEF FN

format: DEF FN (function name)

((parameter list))=(function definition)

function: To define and name a function that is written by user.

END

format: END

function: To terminate program execution, close all files and return to command level

ERROR

format: ERROR (integer expression)

function: To simulate the occurrence of an error or to allow error codes to be defined by the user.

FOR—NEXT

format: FOR (variable)=(initial value) to (final value) (STEP (increment)) NEXT ((variable)), (, (variable) ...)

function: To allow a series of instructions to be performed in a loop a given number of times.

GOSUB—RETURN

format: GOSUB (line number)
RETURN ((line number))

function: To branch to subroutine beginning at (line number) and return from a subroutine.

GOTO

format: GOTO (line number)

function: To branch unconditionally out of the normal program sequence to a specified (line number)

IF—THEN/IF—GOTO

format: IF (expression) THEN (statement) (line number) or IF (expression) GOTO (line number)

function: To make a decision regarding program flow based on the result returned by an expression.

INPUT

format: INPUT ("(prompt string)"); (list of variables)

function: To allow input from the keyboard during program execution.

KEY LIST

format: KEY (function key), (string expression)

function: To set a string to specified function key.

format: KEY LIST.

function: To list the contents of all function keys.

LINE INPUT

format: LINE INPUT ("(prompt string)"); (string variable)

function: To input a entire line (up to 254 characters) to a string variable, without the use of delimiters.

LET

format: (LET) (variable)=(expression)

function: To assign value of an expression to a variable

MAX FILES

format: MAX FILES=(expression)

function: To specify the maximum number of files opened at a time. (expression) can be in the range of 0...15. When 'MAXFILES=0' is executed, only SAVE and LOAD can be performed.

ON ERROR GOTO

format: ON ERROR GOTO (line number) To enable error trapping and specify the first line of the error handling subroutines.

ON GOTO/GOSUB

format: ON (expression) GOTO (list of line number) ON (expression) GOSUB (list of number)

function: To branch to one of several specified line numbers, depending on the value returned when an expression is evaluated.

—ON/OFF/STOP

format: KEY (function key) ON/OFF/STOP

function: To activate/deactivate trapping of the specified function key in a BASIC program.

format: STRIG ((n)) ON/OFF/STOP

function: To activate/deactivate trapping of the trigger buttons of joy sticks in a BASIC program

format: STOP (ON/OFF/STOP)

function: To activate/deactivate trapping of a control-STOP.

format: STRITE (ONE/OFF/STOP)

function: To activate/deactivate trapping of time interval in a BASIC program

ON — GOSUB

format: ON KEY GOSUB (list of line number)

function: To set up a line numbers for BASIC to trap to when the function keys are pressed

format: ON STOP GOSUB (line number)

function: To set up a line numbers for BASIC to trap to when the control-STOP key is pressed

format: ON STRIG GOSUB (line number)

function: To set up a line numbers for BASIC to trap to when the trigger button is pressed.

format: ON SPRITE GOSUB (line number)

function: To set up a line number for BASIC to trap to when the sprites coincide.

format: ON INTERVAL = (time interval) GOSUB (line number)

function: To set up a line number for BASIC to trap to time interval.

OPEN

format: OPEN "(device descriptor) ((file name))" (FOR INPUT/OUTPUT) AS () (file number)

function: This statement opens a device for further processing. Currently, following devices are supported.

CAS: Cassette

CRT: CRT screen

LPT: Line printer.

KBD: Key board

OUT

format: OUT (integer expression) integer expression 2)

function: To send a byte to machine output part (integer expression) is the data (byte) to be transmitted.

POKE

format: POKE (address of the memory) (integer expression)

function: To write a byte (data) into the memory location.

PRINT/LPRINT

format: PRINT (list of expressions)

LPRINT (list of expressions)

function: To output data to the console/printer

PRINT/LPRINT USING

format: PRINT USING (string expression); (list of expression)

LPRINT USING (string expression); (list expression)

function: To print strings or numerics using a specified format to the console printer

PRINT /INPUT

format: PRINT (file number), (exp)

INPUT (file number), (variable list)

function: To write/read data to/from the specified channel.

READ

format: READ (list of variables)

function: To read values from a DATA statement and assign them to variables.

REM

format: REM (remark)

function: To allow explanatory remarks to be inserted in a program

RESTORE

format: RESTORE (line number)

function: To allow DATA statement to be re-read from a specified line.

RESUME

format: RESUME

RESUME NEXT

RESUME (line number)

function: To continue program execution after an error recovery procedure has been performed.

STOP

format: STOP

function: To terminate program execution and return to command level.

TIME

format: TIME = (type:unsigned integer)

function: The system internal timer TIME is automatically incremented by 1 everytime VDP generate interrupt (60 times per second)

The Statement Regarding Graphic and Sound

BEEP

format: BEEP

function: To generate a beep sound

CIRCLE

format: CIRCLE (Coordinate specifier radius) (, (color)), ((start angle)) (, (end angle)) (, (spect ratio))

function: To draw an ellipse with a center and radius as indicated by the first of its arguments.

COLOR

format: COLOR ((foreground color)), (, (background color)) (, (border color))

function: To define the color.

Defaults to 15, 4, 7 acutal color corresponding to each value is as follows:

0. transparent
1. black
2. medium green
3. light green
4. dark blue
5. light blue
6. dark red
7. cyan
8. medium red
9. light red
10. dark yellow
11. light yellow
12. dark green
13. magenta
14. gray
15. white

DRAW

format: DRAW (string expression)

function: To draw figure according to the graphic macro language.

GET/PUT

format: GET (X1 coordinate), (Y1 coordinate), —(X2 coordinate), (Y2 coordinate) (Arrayname)

PUT (X coordinate) (Y coordinate), (array name), (Option)

function: To set/reset the pattern of on the graphic screen into array option is PSET, PRESET, AND, OR, XOR

LINE

format: LINE ((X coordinate), (Y coordinate)) —((X coordinate), Y coordinate)) (, (color)) (,B/BF)

function: To draw line connecting the two specified coordinate

LOCATE

format: LOCATE ((X)) ((XY)) (, (cursor display switch))

function: To locate character position for PRINT

PUT SPRITE

format: PUT SPRITE (sprite plane number) (, (coordinate specifies) (, (color)) (, (pattern number))

function: To set up sprite attribute. X coordinate (X) may range from -32 to 255. Y coordinate (Y) may range from -32 to 191.

If 208 (& HD0) is given to (Y), all sprite planes behind disappears until a value other than 208 is given to that plane. If 209 (& HD1) is specified to (Y), then that sprite disappears from the screen.

PAINT

format: PAINT (coordinate specifies) (, (paint color)) (, (color regarded as border))

function: To fill in an arbitray graphics figure of the specified fill color starting at (coordinate specifier)

PLAY

format: PLAY (string exp for voice 1) (, (string exp for voice 2)) (, (string exp for voice 3)))

function: To play music according to music macro language.

PSET/PRESET

format: pset ((X coordinate), (Y coordinate) (, (color)) PRESET (coordinate specifier) (, (color)).

function: To set/reset the specifier coordinate.

SCREEN

format: SCREEN ((mode)), (, (sprite size)) (, (printer option))

function: To assign the screen mode, sprite size, key click, cassette baud rate and printer option.

SCREEN MODE (0): 40x24 text mode

Screen mode (0): 40x24 text mode

(1): 32x24 text mode

(2): high resolution

mose

(3): multi color mode

Sprite size (0): 8x8 unmagnified

(key click switch) determines whether to enable or disable the key click.

SPRITE \$

format: SPRITE \$ ((pattern number) = (string line))

function: The pattern of sprite (pattern number) must be less than 256 when size of sprite is 0 or 1, less than 64 when size of sprites is 2 to 3. The length of this variable is fixed to 32 (bytes)

VPOKE

format: (address of VRAM), (value to be written)

function: To poke a value to specified location of VRAM. (address of VRAM) can be in the range of 0 ... 16383. (Value to be written) should be a byte value (in the range of 0 ... 255)

- ON/OFF

format: CLICK (ON/OFF)

function: To determine whether the click sound is generated or not, when the key is pressed.

format: MOTOR (ON/OFF)

function: To change the states of cassette motor switch.

format: SOUND (ON/OFF)

function: To write value directly, to the (register of PSG)

SOUND

format: SOUND (register of PSG), (value to be writtern)

function: To write value directly to the (the register of PSG)

WIDTH

format: WIDTH (width of screen in text mode)

function: To set the width of display during text mode. Legal value is 1 ... 40. is 40x24 text mode, 1 ... 32x24 text mode

FUNCTIONS

BIN \$

format: BIN\$(n)

function: Returns a string which represents the binary value of the decimal arguments.

POINT

format: POINT (X coordinate), (Y coordinate)

function: Returns color of a specified pixel.

VPEEK

format: vpeek ((address of RAM))

function: Returns a value of VRAM specified (address) of VRAM can be in the range of 0 ... 16383.

STICK

format: STICK (n)

function: Returns the direction of a joy-shick, (n) can be in the range of 0 to 2, If (n) = 0, the cursor key is used as a joy-stick. If (n) is either/or 2, the joy-stick connected to proper port is used. When neutral, 0 is returned. Otherwise, value corresponding to direction is returned.

STRIG

format: STRIG (n)

function: Returns the states of a trigger button of a joy-stick. (n) can be in the range of 0...4. If (n) = 0, the space bar is used for a trigger button. If (n) is either 1 or 3, the trigger of used for a trigger button. When (n) is either 2 or 4, joy-stick 3. 0 is returned if the trigger is not pressed, -1 is returned otherwise

PDL

format: PDL (n)

function: Returns the value of a paddle. (n) can be in the range of 1...12. When (n) is either 1, 3, 5, 7, 9 or 11, the paddle connected to port 1 is used. When 2, 4, 6, 8, 10 or 12, the paddle connected to part 2 is used.

PAD

format: PAD (n)

function: Returns various status of touch pad. (n) can be in the range of 0-7. When 0...3 is specified, touch pad connected to joy stick part 1 is selected, when 4...7, part 2. When (n) = 0 or 4, the status of touch pad is returned, -1 when touched, 0 when released. When (n) = 1 or 5, the X-coordinate is returned, when (n) = 2 or 6, Y-coordinate is returned. When (n) = 3 or 7, the status of switch on the pad is returned -1 when being pushed, 0 otherwise.

PLAY

format: PLAY (string exp for voice 1) ((string exp for voice 2) ((string exp for voice 3))

function: (string exp for voice n) is a string expression consisting of single character music commands. When a null string is specified, the voice channel remains silent. In the process of playing music, the value of the expression may range from 0 to 3. The value of 3 is to modulate all the channel.

TIME

format: TIME

function: It shows the system internal timer.

The Mathematical Function, Character Function and Special Function

ABS(X)

Returns the absolute value of the expression X.

ASC(XS)

Returns a numerical value that is the ASCII code of the first character of the string XS. If XS is null, a 'illegal function call' error is returned.

ATN(X)

Returns the arctangent of X in radians. Result is in the range $-\pi/2$ to $\pi/2$. The expression X may be any numeric type, but the evaluation of ATN is always performed in double precision.

BINS(n)

Returns a string which represents the binary value of the decimal argument. n is a numeric expression in the range -32768 to 65535. If n is negative, the two's complement form is used. That is, BINS(-n) is the same as BINS(65535-n)

CDBL(X)

Converts X to a double precision number.

CHR\$(I)

Returns a string whose one element is the ASCII code for I. ASCII is commonly used to send a special character to the console, etc.

CINT(X)

Converts X to a integer number by truncating the fractional portion. If X isn't the range -32768 to 32767, an 'Overflow' error occurs.

COS(X)

Returns the cosine of X in radians. COS(X) is calculated to double precision.

LOG(X)

Returns the natural logarithm of X. X must be greater than zero.

LPOS(X)

Returns the current position of the line printer head within the line printer buffer. Does not necessarily give the physical position of the print head. X is a dummy argument.

MID\$(X\$, I, J)

Returns a string of length J characters from X\$ beginning with the Ith character. I and J must be in the range 1 to 255. If J is omitted or if there are fewer than J characters to the right of the Ith character, all rightmost characters beginning with the Ith character are returned. If I = LEN(S\$), MID\$ returns a null string.

OCT\$(n)

Returns a string which represents the octal value of the decimal argument. n is a numeric expression in the range -32768 to 65535. If n is negative, the two's complement form is used. That is, OCT\$ (-n) is the same as OCT\$ (65536-n).

PEEK(I)

Returns the byte (decimal integer in the range 0 to 255) read from memory location I. I must be in the range -32768 to 65535. PEEK is the complementary function to the POKE statement.

POS(I)

Returns the current cursor position. The leftmost position is 0. I is a dummy argument.

RIGHT\$(X\$, I)

Returns the rightmost I characters of string X\$. If I = LEN(X\$), return X\$. IF I = 0, a null string (length zero) is returned.

RND(X)

Returns a random number between 0 and 1. The same sequence of random number is generated each time the program is RUN. If X > 0, the random generator is

reseeded for any given X. X = 0 generates the next random number in the sequence. next random number in the sequence.

SGN(X)

Returns 1 (for X > 0), 0 (for X = 0), -1 (for X < 0)

SIN(X)

Returns the sine of X in radians. SIN(X) is calculated to double precision.

SPACE\$(X)

Returns the string of spaces of length X. The expression X discards the fractional portion and must be range 0 to 255.

SPC(I)

Prints I blanks on the screen. SPC may only be used with PRINT and LPRINT statements. I must be in the range 0 to 255.

SQR(X)

Returns the square root of X. X must be greater than zero.

STR\$(X)

Returns a string representation of the value of X.

STRING\$(I, J)**STRING\$(I, X\$)**

Returns a string of length I whose characters all have ASCII code J or the first character of the string X\$.

TAB(I)

Spaces to position I on the console. If the current print position is already beyond space I, TAB does nothing. Space 0 is the leftmost position, and the rightmost position is the width minus one. I must be in the range 0 to 255. TAB may only be used with PRINT and LPRINT statements.

TAN(X)

Returns the tangent of X in radians. TAN(X) is calculated to double precision. If TAN overflows, an 'Overflow' error will occur.

USR((digit)) (X)

Calls the user's assembly language

CSNG(X)

Converts X to a single precision number.

CSRLIN

Returns the vertical coordinate of the cursor.

ERL/ERR

When an error handling subroutine is entered, the variable ERR contains the error code for error, and the variable ERL contains the line number of the line in which the error was detected. The ERR and ERL variables are usually used in IF ... THEN statements to direct or program flow in the error trap routine. If the statement that caused the error was a direct mode statement, ERL will contain 65535. To test if an error occurred in a direct statement, use IF 65535 = ERL THEN

Otherwise, use

IF ERL = (line number) THEN

IF ERR = (error code) THEN

Because ERL and ERR are reserved variables, neither may appear to the left of the equal sign in a LET (assignment) statement.

EXP(X)

Returns e to the power of X. X must be (-145.06286085862). If EXP overflows, the 'Overflow' error message is printed.

FIX(X)

Returns the integer part of X (fraction truncated). FIX(X) is equivalent to SGN(X) * INT(ABS(X)). The major difference between FIX and INT is that FIX does not return the next lower number for negative X.

FRE(0)**FRE (" ")**

Arguments to FRE are dummy arguments. FRE returns the number of bytes in memory not being used by BASIC.

FRE(0) returns the number of bytes in memory which can be used for BASIC program, text file, machine language program file, etc. FRE(" ") returns the number of bytes in memory for string space.

HEX\$(X)

Returns a string which represents the hexadecimal value of the decimal argument.

n is a numeric expression in the range -32768 to 65535. If n is negative, the two's complement form is used. That is, HEX\$(-n) is the same as HEX\$(65536-n).

INKEY\$

Returns either a one-character string containing a character read from the keyboard or a null string if no key is pressed. No characters will be echoed and all characters are passed through to the program except Control-C, which terminates the program.

INPUT\$(X)

Returns a string of X characters, read from the keyboard. No character will be echoed and all characters are passed through except Control-C, terminates the execution of the INPUT\$ function.

INSTR (I, X\$, Y\$)

Searches for the first occurrence of string Y\$ in X\$ and returns the position at which the match is found. Optional offset I sets the position for starting the search. I must be in the range 0 to 255. If I = LEN(X\$) or if S\$ is null or if Y\$ cannot be found or if S\$ and Y\$ are null, INSTR returns 0. If only Y\$ is null, INSTR returns I or 1. X\$ and Y\$ may be string variables, string expressions, or string literals.

INT(X)

Returns the largest integer S smaller than X

LEFT\$(X\$, I)

Returns a string comprising the leftmost I characters of X\$. I must be in the range 0 to 255. If I is greater than LEN(X\$), the entire string (X\$) is returned.

IF I = 0, a null string (length zero) is returned.

LEN(X\$)

Returns the number of characters in X\$. Nonprinting characters and blanks are counted.

subroutine with the argument X. (digit) is in the range 0 to 9 and corresponds to the digit supplied with the DEFUSR statement for that routine. If (digit) is omitted, USR0 is assumed.

VAL(X\$)

Returns the numerical value of the string X\$. The VAL function also strips leading blanks, tabs, and linefeeds from the argument string. For example

```
PRINT VAL (" -7")
```

```
-7
```

```
OK
```

VARPTR((file number))

Returns the address of the first byte of data identified with (variable name). A value must be assigned to (variable name) prior to execution of VARPTR. Otherwise, an 'Illegal

function call' error results. Any type variable name may be used (numeric, string, array), and the address returned will be an integer in the range -32768 to 32767. If a negative address is returned, add it to 65536 to obtain the actual address.

VARPTR is usually used to obtain the address of a variable or array so it may be passed to a machine language subroutine. A function call of the form VARPTR (A(0)) is usually specified when passing an array, so that the lowest address element of the array is returned.

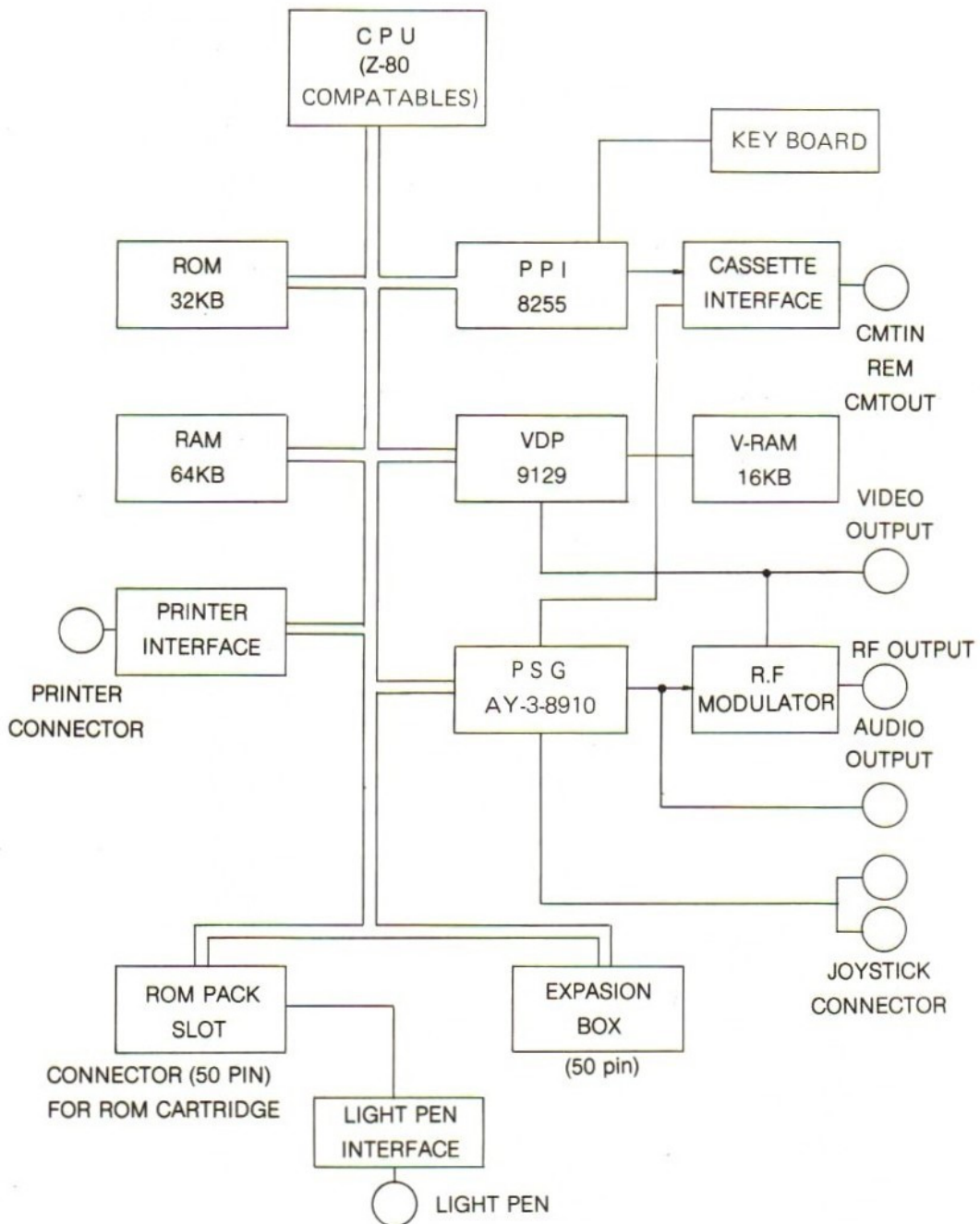
All simple variables should be assigned before calling VARPTR for an array because the address of the arrays change whenever a new simple variable is assigned. If (file number) is specified, VARPTR returns the starting address of the file control block.

16. APPENDIX

1. SPECIFICATION OF THE PRODUCT

C P U		Z-80A (3.58MHz)
MEMORY	ROM	32KB (BASIC INTERPRETER)
	RAM	80KB (INCLUDING VRAM)
DISPLAY CAPACITY	TEXT MODE	40 columnx24 row 32 columnx24 row • 16 COLOR
	HIGH RESOLUTION GRAPHIC MODE	256 dotx192 dot • 16 COLOR
	LOW RESOLUTION GRAPHIC MODE	64 blockx48 block (4x4 dot/block) • 16 COLOR
CONNECTION TERMINAL	VIDEO OUTPUT	Terminal/Composite video signal RCA PIN
	RF OUTPUT	Terminal/channel UHF36 (including sound signal) • RCA PIN
	AUDIO OUTPUT	Terminal /;8 octave-3 tone-RCA PIN
	CASSETTE OUTPUT	Terminal/FSK system • 1200/2400 Baud rate 8 pin DIN connector
	PRINTER OUTPUT	Terminal/Centronics parallel amphenol 14 pin
	JOY STICK	Terminal 2/9 PIN D-type connector
	SLOT	2 SOLT/50 PIN card edge connector 50 PIN expansion slot connector
KEY BOARD		73 KEY/English, Graphic key
POWER		AC 240V available (50/60Hz)
POWER DISSIPATION		20W
HOUSING DIMENSION		400 (Horizontal)x265.5 (vertical)x 63(height) (Unit: mm)
WEIGHT		4.7 kg
TEMPERATURE AVAILABLE		0°C—35°C

2. BLOCK DIAGRAM



• ROM CARTRIDGE EXPANSION BUS CONNECTOR SIGNAL

No.	Name	I/O*	No.	Name	I/O*
1	CS1	O	2	CS2	O
3	CS12	O	4	SLTSL	O
5	Reserved*1	—	6	RFSH	O
7	WAIT *2	I	8	INT *2	I/O
9	M1	O	10	BUSDIR	I/O
11	IORQ	O	12	MERQ	O
13	WR	O	14	RD	O
15	RESET	O	16	Video signal	O
17	A9	O	18	A15	O
19	A11	O	20	A10	O
21	A7	O	22	A6	O
23	A12	O	24	A8	O
25	A14	O	26	A13	O
27	A1	O	28	A0	O
29	A3	O	30	A2	O
31	A5	O	32	A4	O
33	D1	I/O	34	D0	I/O
35	D3	I/O	36	D2	I/O
37	D5	I/O	38	D4	I/O
39	D7	I/O	40	D6	I/O
41	GND	—	42	CLOCK	O
43	GND	—	44	SW1	—
45	+5V	—	46	SW2	—
47	+5V	—	48	+12V	—
49	SUNDIN	I	50	-12V	—


* The difference between input and output based on the system

*1: "Reserved" means that this pin is prohibited to be used.

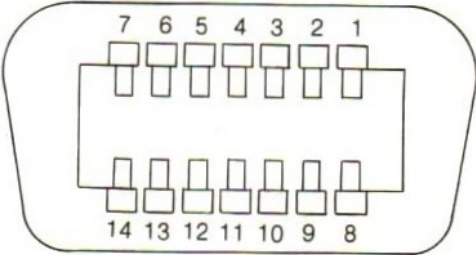
*2: OPEN COLLECTOR OUTPUT

No.	Name	Contents
1	CS1	ROM 4000H—7FFFH Address select signal
2	CS2	ROM 8000H—BFFFH Address select signal
3	CS12	ROM 4000H—BFFFH Address select signal (for 256 ROM)
4	SLTSL	Slot select signal the unique of signal of the slot is applied to each slot.
5	RESERVED	
6	RFSH	Refresh cycle signal
7	WAIT	Wait request signal to CPU.
8	INT	Interrupt request signal to C.P.U
9	MI	The signal showing the fetch cycle of C.P.U
10	BUSDIR	The signal controlling the external data bus At the timing that the contridge is selected and data are transmitted, it divides the memory and outputs the lower level than the cartridge
11	IORQ	I/O request signal
12	MERQ	Memory request signal
13	WR	Write timing signal
14	RD	Read timing signal
15	RESET	System reset signal
16	VIDEO	Video signal
17-32	A0-A15	Address bus signal
33-40	D0-D7	Data bus signal
41	GND	Signal ground
42	CLOCK	CPU clock (3.58 MHz)
43	GND	Signal ground
44, 46	SW1, SW2	For the protection of the system
45, 47	+5V	+5V power
48	+12V	+12V power
49	SUNDIN	sound input signal (-5 dbm)
50	-12V	-12V power

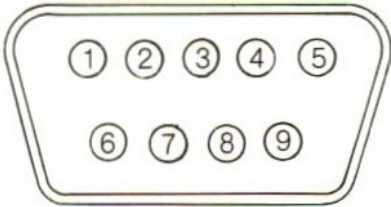
• CASSETTE INTERFACE CONNECTOR

Terminal number	Signal name	Direction	pin configuration
1	GND	—	 <p>Pin arrangement toward the system</p>
2	GND	—	
3	GND	—	
4	CMTOUT	OUTPUT	
5	CMTIN	INPUT	
6	REM +	OUTPUT	
7	REM-	OUTPUT	
8	GND		

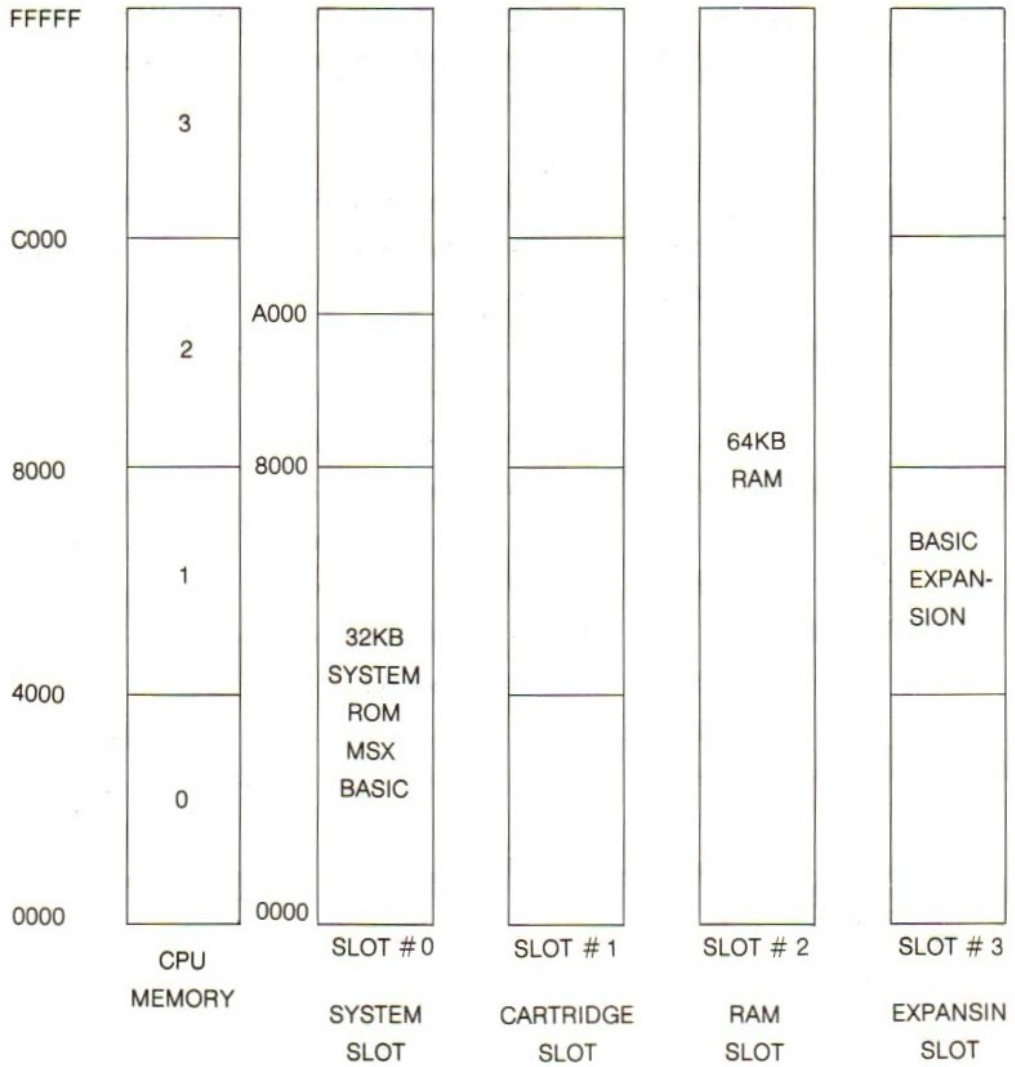
• PRINTER INTERFACE CONNECTION

Terminal number	Signal name	pin configuration
1	PSTB	 <p>pin arrangement toward the system</p>
2	PD0	
3	PD1	
4	PD2	
5	PD3	
6	PD4	
7	PD5	
8	PD6	
9	PD7	
10	NC	
11	BUSY	
12	NC	
13	NC	
14	GND	

• JOY STICK INTERFACE CONNECTION

Terminal number	Signal name	Direction	pin configuration
1	FWD	INPUT	 <p>pin arrangement toward the system</p>
2	BACK	INPUT	
3	LEFT	INPUT	
4	RIGHT	INPUT	
5	+ 5V	—	
6	TRG1	IN, OUT	
7	TRG2	IN, OUT	
8	OUTPUT	OUTPUT	
9	GND	—	

3. MEMORY MAP



4. I/O MAP

I/O ADRS DEVICE

FF		I/O ADR	RW	Contents	Reference
E0		&H98	W	Write data to V-RAM	TMS9129NL (or compatible chip)
D8	*FDC	&H99	R	Read data from V-RAM	
D0			W	Command, Address set	
			R	Status read	
C0	SYSTEM RESERVE AREA	&HA0	W	Address latch	AY-3-8910 (or compatible chip)
		&HA1	W	Data write	
		&HA2	R	Data read	
B0	PPI	&HA8	W	Port a data write	8255A (or compatible chip)
			R	Port a data read	
A8	PSG	&HA9	W	Port B data write	
			R	Port b data read	
A0	VDP	&HAA	W	Port c data write	
			R	Port c data read	
		&HAB	W	Mode set	
98	PRINTER	&H90	W	Strobe output	latch output BUSY'1' latch output
		R	R	Status input	
		&H91	W	Print data	
	SYSTEM RESERVE AREA	&H80	W	Data write	8251A (or compatible chip)
	*RS-232C		R	Data read	
80	UNDEFINED	&H81	W	Mode set	
			R	Status read	

* (Asterisk) means the use of expansion devices.

5. CONTROL KEY TABLE

Table 1. MSX BASIC control functions. The ASCII control key is entered by pressing the key while holding down the control key.

Hex. code	Control key	Special key	Function	Control key
01	A		This is used in case of I/O of the graphic character.	CTRL + A
02*	B		Move cursor to start of previous word	CTRL + B*
03*	C		Break when MSX BASIC is waiting for input	CTRL + D*
04*	D		Ignored	CTRL + D*
05*	E		Truncate line (clear text to end of logical line)	CTRL + E*
06*	F		Move cursor to start of next word	CTRL + F*
07*	G		Beep	CTRL + G*
08	H	Back space	Backspace, deleting characters passed over	CTRL + H, BS
09	I	Tab	Tab (moves to next TAB stop)	CTRL + 1 TAB
0A*	J		Line feed	CTRL + J*
0B*	K	Home	Move cursor to home position	CTRL + K*
0C*	L	CLS	Clear screen	CTRL + L*
0D*	M	Return	Carriage return (enter current logical line)	CTRL + M, RETURN*
0E*	N		Append to end of line	CTRL + N*
0F*	O		Ignored	CTRL + O*
10*	P		Ignored	CTRL + P*
11*	Q		Ignored	CTRL + Q*
12*	R	INS	Toggle insert/typeover mode	CTRL + R, INS*
13*	S		Ignored	CTRL + S*
14*	T		Ignored	CTRL + T*
15*	U		Clear logical line	CTRL + U*
16*	V		Ignored	CTRL + V*
17*	W		Ignored	CTRL + W*
18*	X	Select		CTRL + X, SELECT*
19*	Y		Ignored	CTRL + Y
1A*	Z		Ignored	CTRL + Z
1B	(ESC	Ignored	CTRL + (, ESC
1C*	W	Right arrow	Cursor right (W is Won sign)	CTRL + W, —*
1D*)	Left arrow	Cursor left	CTRL +), —8
1E*	:	Up arrow	Cursor down	CTRL + ;, 4*
1F*	—	Down arrow	Cursor down	CTRL + SHIFT + —*
7F	DEL	DEL	Delete character at cursor	DEL

Note: Those keys marked with asterisk (*) cancels insert mode when editor is in insert mode.

6. MSX CODE TABLE

		Upper 4 Bit																		
		0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1																		
		0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1																		
		0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1																		
Lower 4 Bit		0 1 2 3 4 5 6 7 8 9 A B C D E F																		
0	0	0	0	0	Blank (NULL)	+	Blank (Space)	θ	@	P	'	p	ç	É	á	Ã	◀	α	≡	
0	0	0	1	1	☺		!	1	A	Q	a	q	Ü	æ	í	ã	⊗	β	±	
0	0	1	0	2	☹		“	2	B	R	b	r	e	Æ	ó	ÿ	⊗	γ	≥	
0	0	1	1	3	♥		#	3	C	S	c	s	â	ô	ú	ÿ	⊗	π	≤	
0	1	0	0	4	♦		\$	4	D	T	d	t	ã	õ	ñ	Õ	■	Σ	⌋	
0	1	0	1	5	♣		%	5	E	U	e	u	à	ð	Ñ	õ	■	σ	⌋	
0	1	1	0	6	♠		&	6	F	V	f	v	â	û	ä	Û	■	μ	÷	
0	1	1	1	7	•		'	7	G	W	g	w	ç	ù	ö	ü	■	τ	≈	
1	0	0	0	8	•		(8	H	X	h	x	e	ÿ	ı	Π	■	∇	φ	°
1	0	0	1	9	○)	9	I	Y	i	y	e	Ö	□	ij	■	‡	•	
1	0	1	0	A	○		*	:	J	Z	j	z	e	Ü	□	¼	■	W	Ω	•
1	0	1	1	B	ó		+	;	K	[k	{	ĩ	φ	½	~	▱	δ	√	
1	1	0	0	C	♀	⊗	,	<	L	\	l		í	£	¼	◇	▱	∞	η	
1	1	0	1	D	♪	⊗	-	=	M]	m	}	ï	≠	i	‰	▱	φ	²	
1	1	1	0	E	♪	⊗	.	>	N	^	n	~	Ä	Pt	<<	¶	▱	€	■	
1	1	1	1	F	☀	+	/	?	O	_	o	△	Å	f	>>	§	▱	∩	Blank (FF)	

7. SUMMARY OF ERROR CODES AND ERROR MESSAGES

Code	Message	Explanation
1	NEXT without FOR	A variable in a NEXT statement does not correspond to any previously executed, unmatched FOR statement variable.
2	Syntax error	A line is encountered that contains some incorrect sequence of characters (such as unmatched parenthesis, misspelled command or statement, incorrect punctuation, etc.)
3	RETURN without GOSUB	A RETURN statement is encountered for which there is no previous, unmatched GOSUB statement.
4	Out of DATA	A READ statement is executed when there are no DATA statement with unread data remaining in the program.
5	Illegal function call	A parameter that is out of the range is passed to a math or string function. An FC error may also occur as the result of: <ol style="list-style-type: none"> 1. a negative or unreasonably large subscript. 2. a negative or zero argument with LOG. 3. a negative argument to SQR. 4. an improper argument to MID\$, LEFT\$, RIGHT\$, INP, OUT, PEEK, POKE, TAB SPC, STRING\$, SPACE\$, INSERTS or ON ... GOTO.
6	Overflow	The result of a calculation is too large to be represented in BASIC's number format.
7	Out of memory	A program is too large, has too many files, has too many FOR loops or GOSUBs, too many variables, or expressions that are too Language specification for MSX BASIC
8	Undefined line number	A line reference in a GOTO, GOSUB, IF ... THEN ... ELSE is to a nonexistent line.
9	Subscript out of range	An array element is referenced either with a subscript that is outside the dimensions of the array, or with the wrong number of subscripts.

Code	Message	Explanation
10	Radimensioned array	Two DIM statements are given for the same array, or DIM statement is given for an array after the default dimension of 10 has been established for that array.
11	Division by zero	A division by zero is encountered in an expression, or the operation of involution results in zero being raised to a negative power.
12	Illegal direct	A statement that is illegal in direct mode is entered as a direct mode command.
13	Type mismatch	A string variable name is assigned a numeric value or vice versa; a function that expects a numeric argument is given a string argument or vice versa.
14	Out of string space	String variables have caused BASIC to exceed the amount of free memory remaining. BASIC will allocate string space dynamically, until it runs out of memory.
15	String too long	An attempt is made to create a string more than 225 character long.
16	String formula too complex	A string expression is too long or too complex. The expression should be broken into smaller expressions.
17	Can't continue	An attempt is made to continue a program that: <ol style="list-style-type: none"> 1. has halted due to an error. 2. has been modified during a break in execution, or Language specification for MSX BASIC. 3. does not exist.
18	Undefined user function	FN function is called before defining it with the DEF FN statement.
19	Device I/O error	An I/O error occurred on a cassette, printer, or CRT operation. It is a fatal error; i.e., BASIC cannot recover from the error.
20	Verify error	The current program is different from the program saved on the cassette.
21	No RESUME	An error trapping routine is entered but contains no RESUME statement.

Code	Message	Explanation
22	RESUME without error	A RESUME statement is encountered before an error trapping routine is entered.
23	Unprintable error	An error message is not available for the error condition which exists. This is usually caused by an ERROR with an undefined error code.
24	Missing operand	An expression contained an operator with no operand following it.
25	Line buffer overflow	An entered line has too many characters.
26	Unprintable errors	These codes have no definitions. Should be reserved for future expansion in BASIC.
49		
50	FIELD overflow	A FIELD statement is attempting allocate more bytes than were specified for the record length of a random file in the OPEN statement. Or, the end of the FIELD buffer is encountered while doing sequential 160 (PRINT , INPUT) to a random file.
51	Internal error	An internal malfunction has occurred. Report to Microsoft the conditions under which the message appeared.
52	Bad file number	A statement or command references a file with a file number that is not OPEN or is out of the range of file number specified by MAXFILE statement.
53	File not found	A LOAD, KILL, or OPEN statement references a file that does not exist in the memory.
54	File already open	A sequential output mode OPEN is issued for a file that is already open; or a KILL is given for a file that is open.
55	Input past end	An INPUT statement is executed after all the data in the file has been INPUT, or for null (empty) file. To avoid this error, use the EOF function to detect the end of file.
56	Bad file name	An illegal form is used for the file name which LOAD, SAVE, KILL, NAME, etc.
57	Direct statement in file	A direct statement is encountered while LOADING an ASCII format file. The LOAD is terminated.

Code	Message	Explanation
58	Sequential I/O only	A statement to random access is issued for a sequential file.
59	File not OPEN	The file specified in PRINT , INPUT , etc. hasn't been OPENED.
60	Unprintable error	These codes have no definitions. Users may place their own error code definitions at the high end of this range.

THE OA PRODUCT INTRODUCTION

MICRO COMPUTER ;

GMC-3010, GMC-3020, GMC-4011, GMC-2010,
GMC-3110, GMC-3030, GMC-2110, GMC-5620,
GMC-6011, GMC-6021, GMC-6031

FAMILY COMPUTER ;

FC-30, FC-80, FC-100, FC-200

DISPLAY TERMINAL ;

GDT-8200, GDT-8100, GDT-7100, GDT-7200, GDT-7210,
GDT-6100

ELECTRONIC CASH REGISTER ;

GCR-204, GCR-210, GCR-304, GCR-310, GCR-404

ELECTRONIC MEMORY TYPEWRITER ;

GTS-8010, GTS-8300, GTS-8300A, GTS-8400

TELLER'S MACHINE ;

GTM-8000A, GTM-8000B, GTM-8000C

PRINTER ;

PRT-5, PRT-10, PRT-20, PRT-30, PRT-6

HEAD OFFICE ; 537, NAMDAEMUN-RO 5GA, JUNG-GU,
SEOUL, KOREA
TEL ; 771-32, TLX ; GSRADIO K23751-5

OA DIVISION ; 459-9, GARIBONG-DONG, GURO-GU,
SEOUL, KOREA
TEL ; 855-9331—5, TLX : GSAUDIO K23373