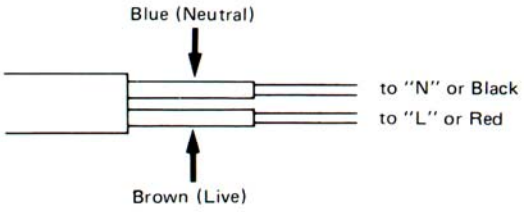
JVC Instructions

PERSONAL COMPUTER

HC-7GB

MSX

POWER CONNECTION



IMPORTANT

The wires in this mains lead are coloured in accordance with the following code:

Blue: Neutral
Brown: Live

As the colours of the wires in the mains lead of this apparatus may not correspond with the coloured markings identifying the terminals in your plug, proceed as follows:

The wire which is coloured blue must be connected to the terminal which is marked with the letter N or coloured black.

The wire which is coloured brown must be connected to the terminal which is marked with the letter L or coloured red.

IDENTIFICATION:

FOR YOUR PROTECTION—
To aid in the identification of this personal computer in the event of theft, etc. please record below the serial number located on the back of the unit. Retain this book for future reference.

MODEL NO. HC-7GB
SERIAL NO. _____
DATE OF PURCHASE _____
DEALER's NAME _____

CONTENTS

INTRODUCTION

Congratulations on your purchase of the JVC MSX personal computer HC-7GB which employs the Microsoft MSX — a standard for software and hardware of 8-bit personal computers.
You can run all programs with the MSX mark available or programs developed with other MSX computers on the HC-7GB. You can also use your programs developed with the HC-7GB on other MSX computers.
Please read this manual carefully and take time to become throughly familiar with your new personal computer.

This manual describes basic operations for using the HC-7GB incorporating procedures on setting up the computer and how to use application programs. It is recommended that you this manual to make the best use of the HC-7GB. See MSX BASIC GUIDE, or instructions commercially available for details on the MSX BASIC.

* MSX is a trademark of Microsoft.

DISCLAIMER

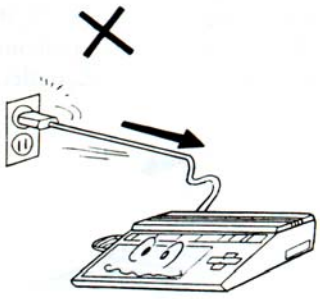
1. Every effort has been made to ensure that the contents of this manual are accurate and complete. However, JVC would appreciate being informed of any errors or ambiguities which may be detected.
2. The above notwithstanding, JVC assumes no responsibility for any errors in this manual or their consequences.

PRECAUTIONS ON USING THE HC-7GB

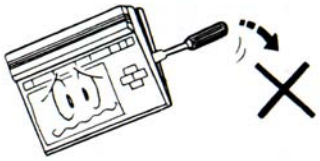
You should observe the following points when using the HC-7GB.

- When something is wrong with the HC-7GB during operation, pull out the AC plug from the AC outlet. It is not recommended that you continue operation with the computer. Contact your Victor dealer as soon as possible.

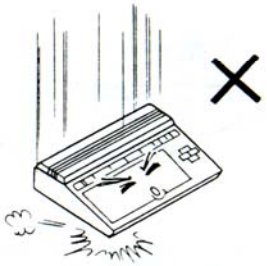
- Do not handle the cables roughly. When disconnecting the AC cable from an AC outlet, be sure to hold the plug and pull it toward you.



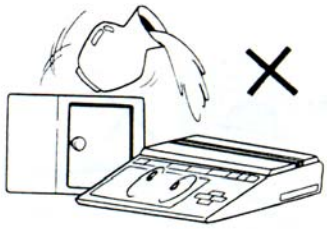
- Do not disassemble or remodel the HC-7GB.



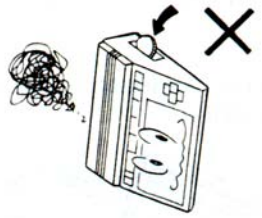
- Do not allow the HC-7GB to be subject to shocks of any kind.



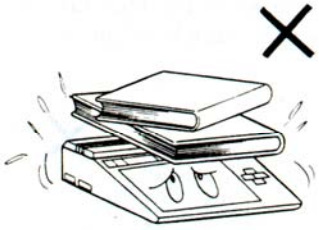
- Avoid using or leaving the HC-7GB in a place exposed to moisture.



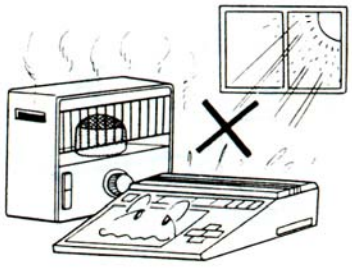
- Do not place any materials especially nails, coins, or flammable object in the ROM cartridge slot.



- Do not place any object on the HC-7GB

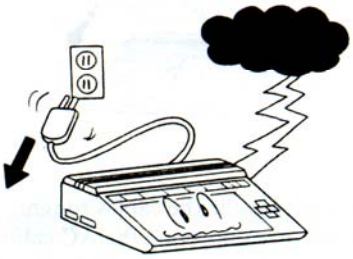


- Avoid using or leaving the HC-7GB in a place exposed to direct sunlight and do not place it in the vicinity of any source of heat.

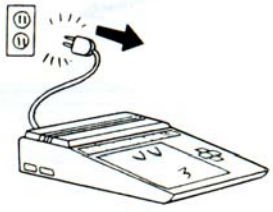


- Avoid using or leaving the HC-7GB in environments which are extremely humid or which contain dust or airborne articles.

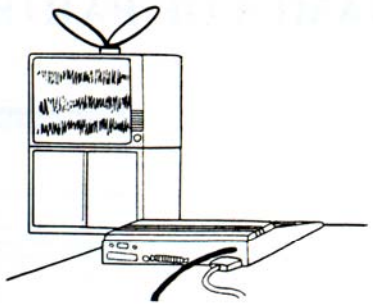
- When you hear a peal of thunder, pull out the AC plug from the AC outlet.



- When you leave the HC-7GB for a long period of time, pull out the AC plug from the AC outlet.



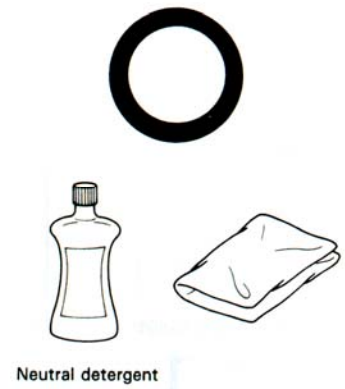
- Using the HC-7GB in the vicinity of a monitor may affect the TV screen image.



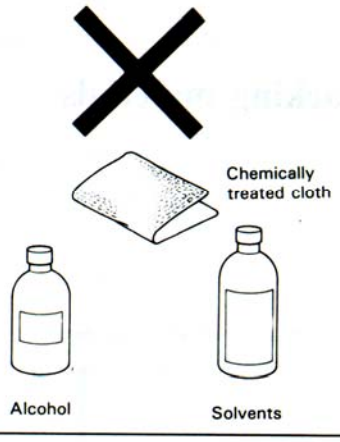
Cleaning

It is recommended that the HC-7GB be cleaned periodically to endure a long operating life. Note the following points when cleaning the HC-7GB.

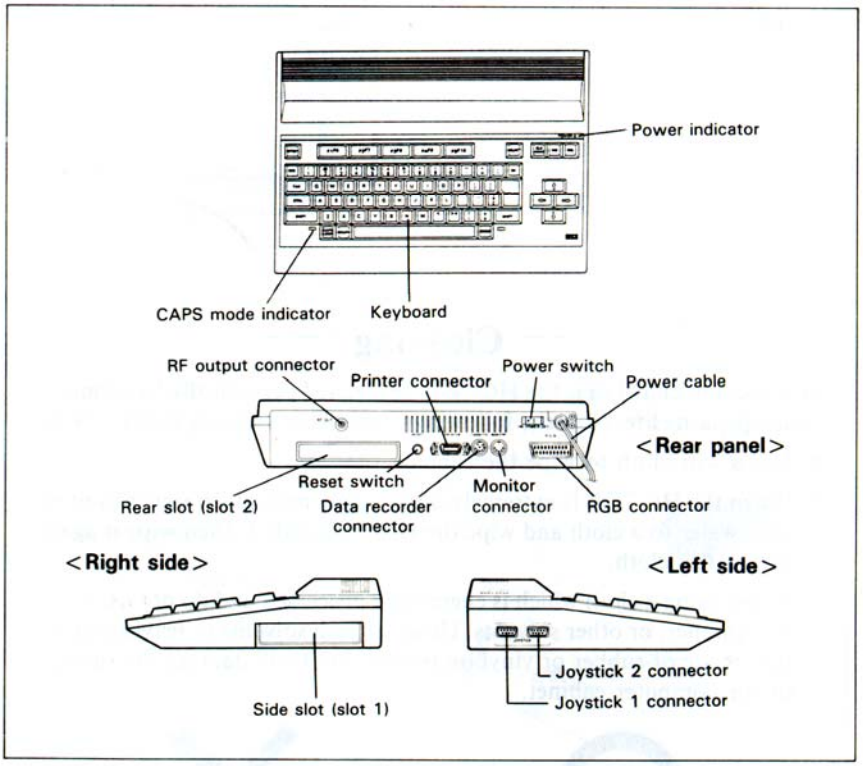
- Use a soft cloth to wipe the cabinet.
- When the HC-7GB is extremely dirty, apply neutral detergent thinned with water to a cloth and wipe the HC-7GB with it; then wipe it again with a dry cloth.
- Avoid using a cloth which is chemically processed and do not use alcohol, thinner, or other solvents. Using of such solvents or leaving an article made of rubber or vinyl on the HC-7GB will damage the surface of the computer cabinet.



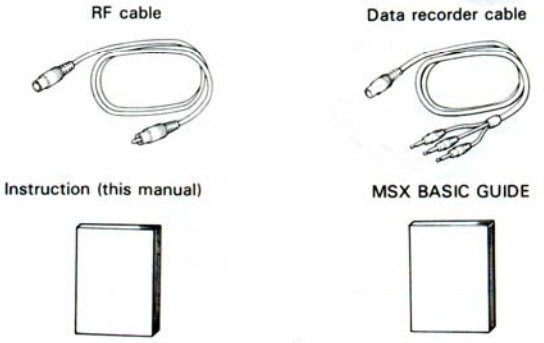
Neutral detergent



NAMES OF PARTS

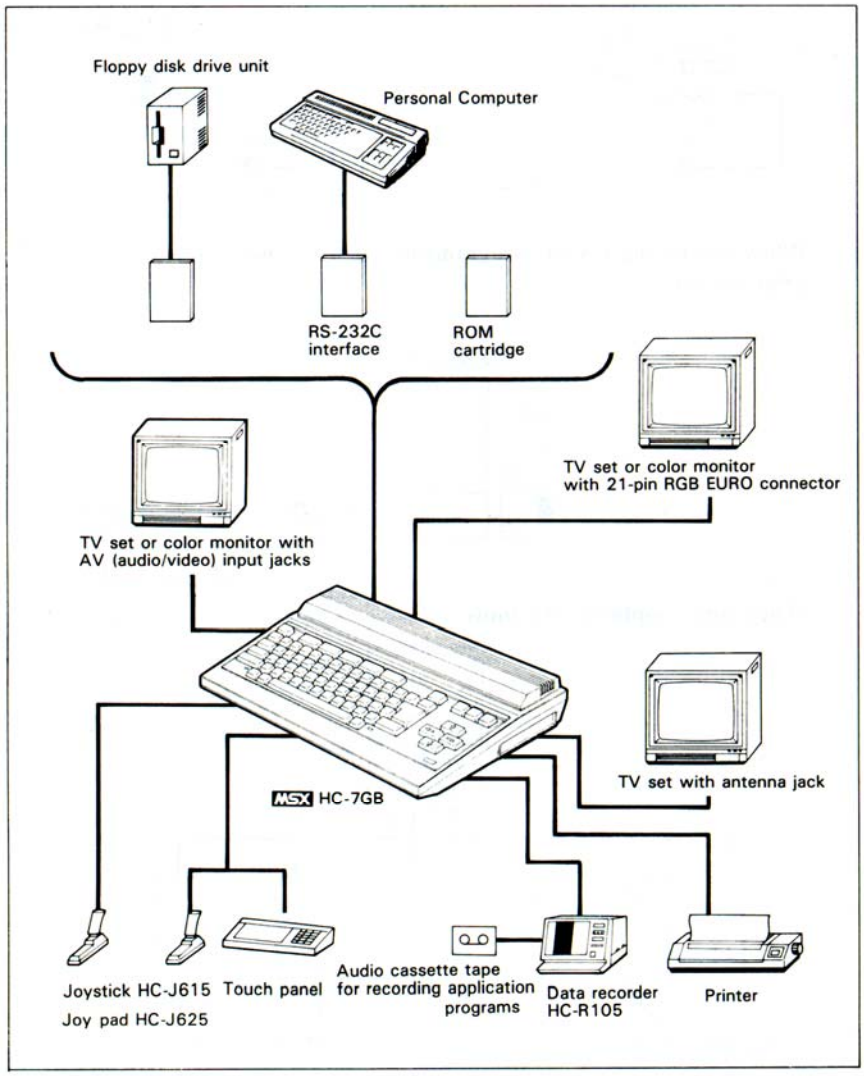


Packing materials



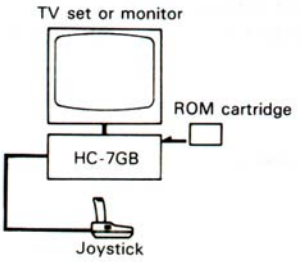
SYSTEM CONFIGURATION

The configuration diagram and system example for each application are shown below. You can set up the system according to your own requirement.

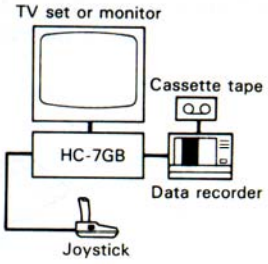


System examples

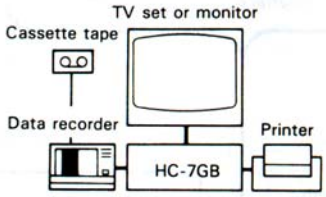
When using a ROM cartridge



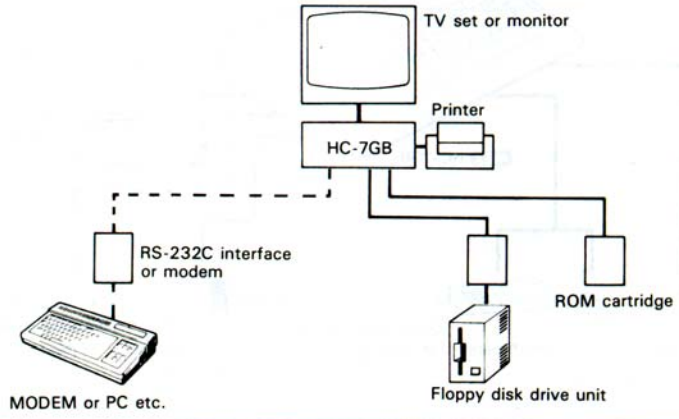
When using a cassette tape



When developing BASIC programs or when it is necessary to save or print out data



When other sophisticated tools



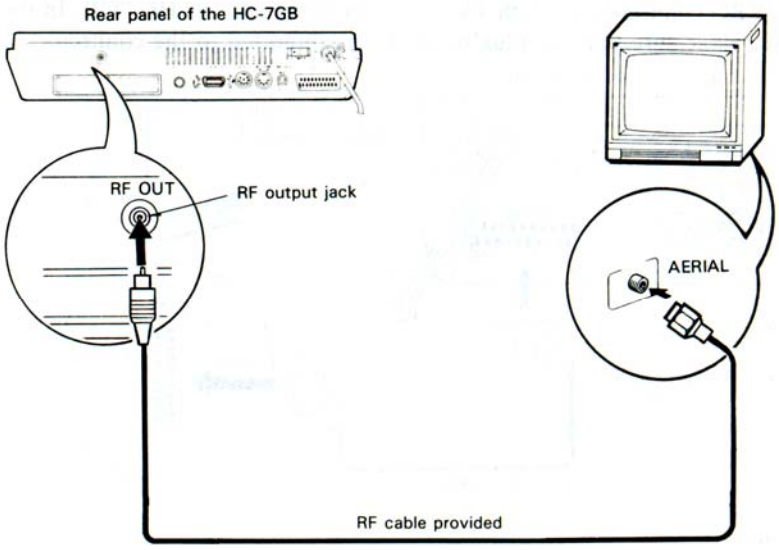
CONNECTION

Be sure to turn off the power switch of the HC-7GB and all peripherals.

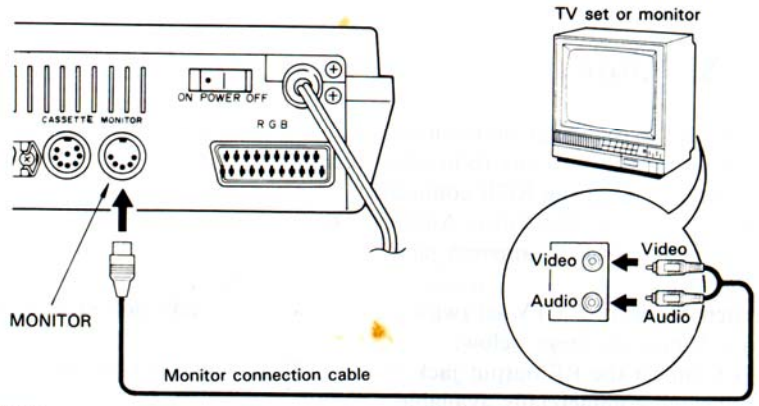
1. Monitor

The clearity of the image on the monitor differs according to the type of connection; it degrades by the following order.
(1) Connection via the RGB connector.
(2) Connection via the Video/Audio connector.
(3) Connection via the antenna jack.

- When connecting a TV set (with neither AV input jacks nor RGB connector), follow the steps below.
  (1) Connect the RF output jack to the UHF antenna jack via an antenna switch commercially available.
  (2) Select TV channel 36.

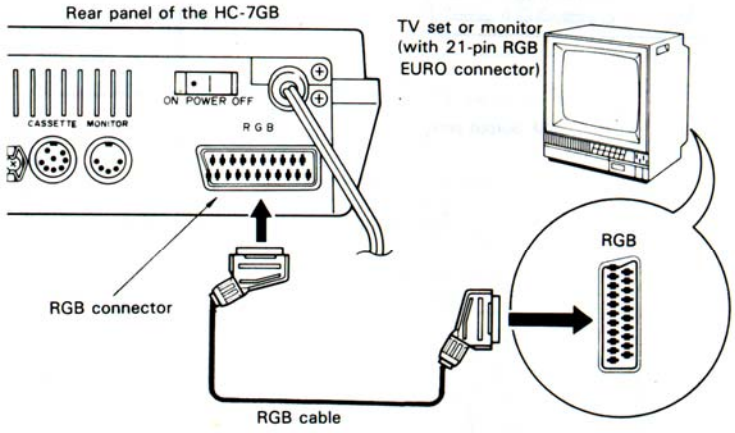


- When connecting a TV set or color monitor with AV input jacks, connect the MONITOR of the HC-7GB and the AV input jacks of the TV set or monitor.



Note:
Use the correct cable for this connector.

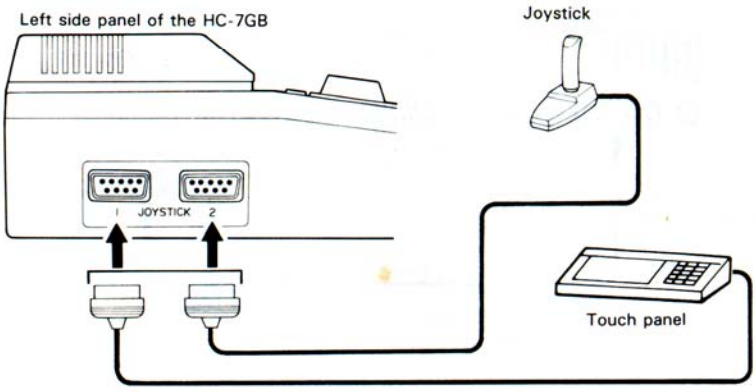
- When connecting a TV set or color monitor with RGB connector, connect the RBG connectors of both TV set (or monitor) and the HC-7GB. In this case, make sure that the plug be properly connected to the connector.



Note:
Use the correct cable for this connector.

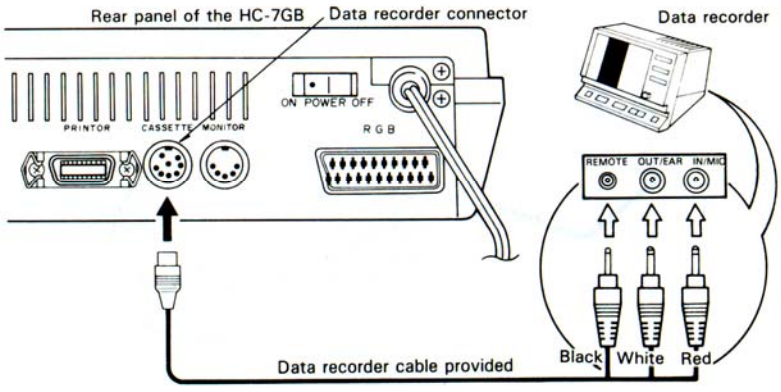
2. Joystick and touch panel

A joystick and touch panel should be connected to the joystick connector 1 or 2. Either connector 1 or 2 is valid according to application programs. If the joystick is connected to an invalid connector, it will not operate.



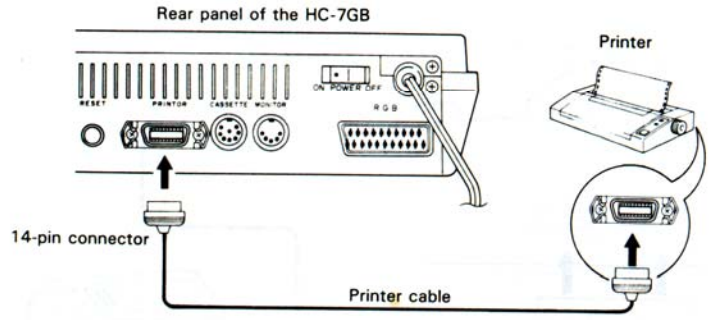
3. Data recorder

Connect the data recorder to the data recorder connector.



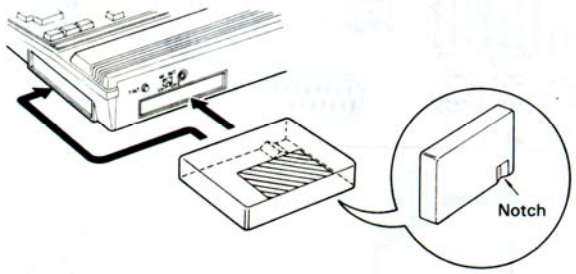
4. Printer

Connect the printer to the PRINTER connector of the HC-7GB. Be sure to fasten the wire clips of both connectors.



5. Cartridges

Insert a ROM cartridge or interface cartridge into the slot at the rear panel or right side panel of the HC-7GB all the way in. The notch side should be underside.



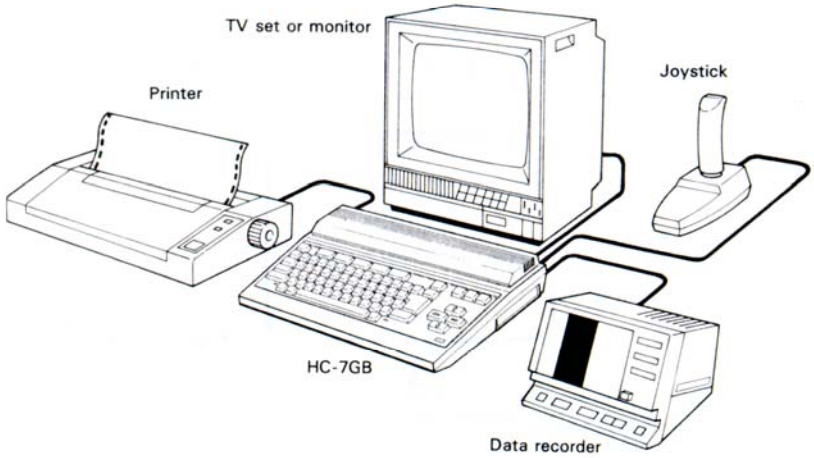
Note:
When cartridges are inserted in both slots, the side slot has the higher precedence.

OPERATION

1. Starting up

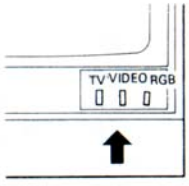
Follow the steps below to turn the power on.

(1) Make sure that monitor, printer, data recorder, joysticks and other peripherals are properly connected.

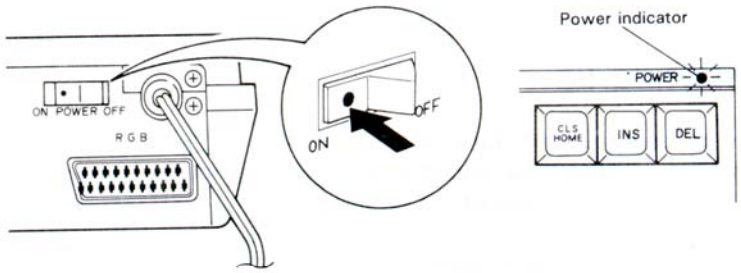


(2) Turn on the power switch of monitor and all peripherals.

(3) When using a TV set, select the setting which displays the computer video output.



(4) Turn the power switch of the HC-7GB on.

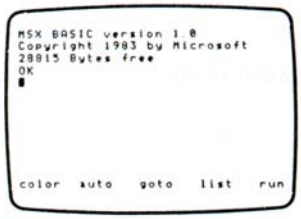


(5) The screen in Figure 1 appears; then, after several seconds, the screen in Figure 2 is displayed.

Fig. 1

After several seconds

Fig. 2



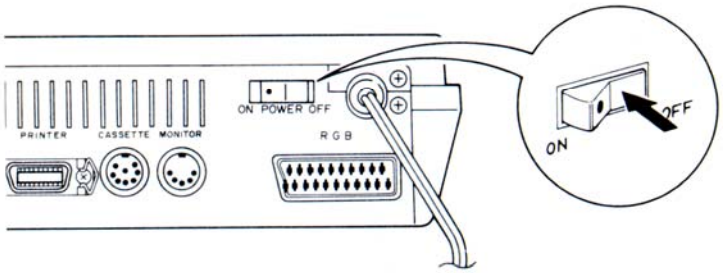
When using a ROM cartridge or floppy disk, the screen differs according to the application programs. See instruction for each unit.
The screen shown in Figure 2 also appears when a Floppy disk unit is connected; however, in this case Disk BASIC is operating.

Notes:

- When the screen is not displayed properly, turn the power switch of the HC-7GB off and turn it back on after several seconds. If normal screen is not displayed with this operation, see "Troubleshooting".
- When the power is supplied to the HC-7GB do not turn the power of peraipherals on or off, or do not connect or disconnect cables. Otherwise, the HC-7GB may malfunction.

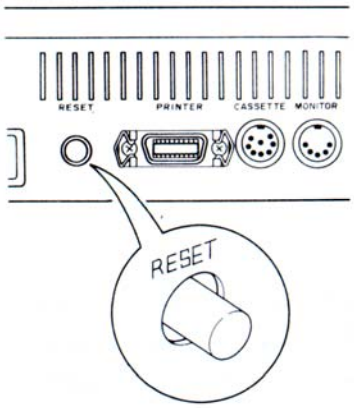
2. Switching the HC-7GB off

Before turning the power of any peripheral, be sure to turn the power switch of the HC-7GB off.

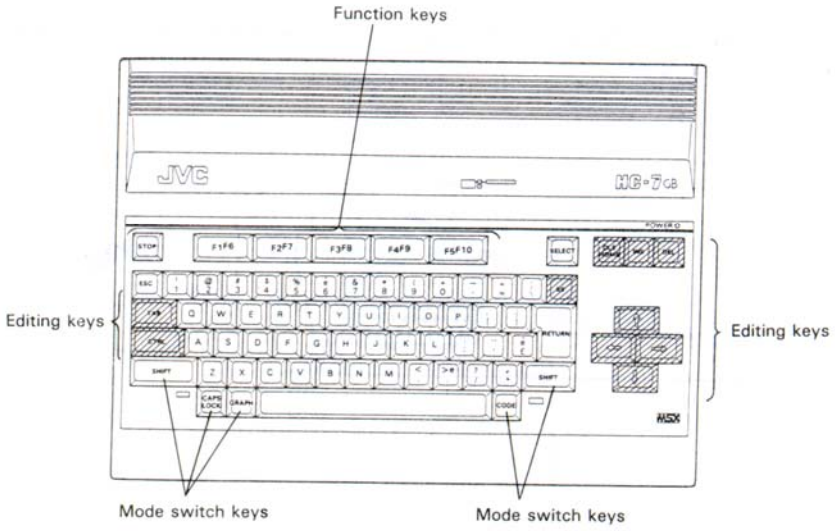


3. Reset sequence

When the HC-7GB malfunctions for any reason, press the reset button to initialize the HC-7GB. When a reset is made, all programs in memory will be lost.



KEYBOARD

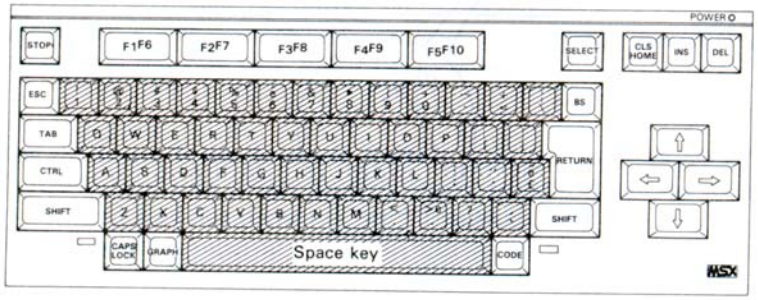


There are 73 keys which can be divided into the following four categories.

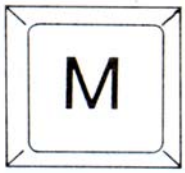
- Character keys
- Mode switch keys
- Editing keys
- Function keys

All character keys and some editing keys are provided with auto-repeat function which makes it possible to enter the same character codes continuously as long as the key is pressed.

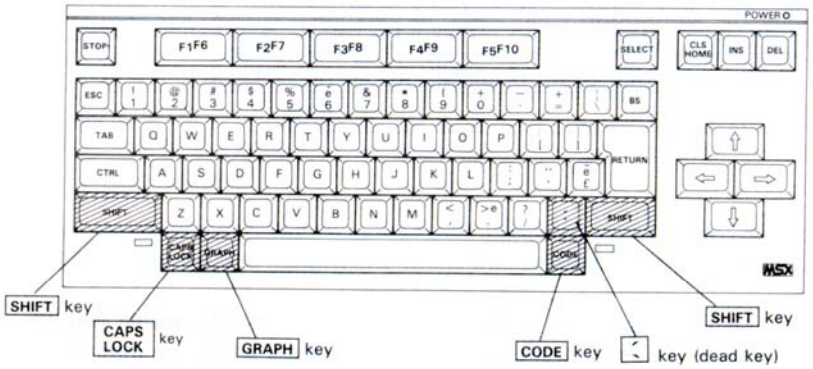
1. Character keys



When a character key is pressed, a character pictured on the key is entered. When the space key is pressed, a space code is entered.



2. Mode switch keys

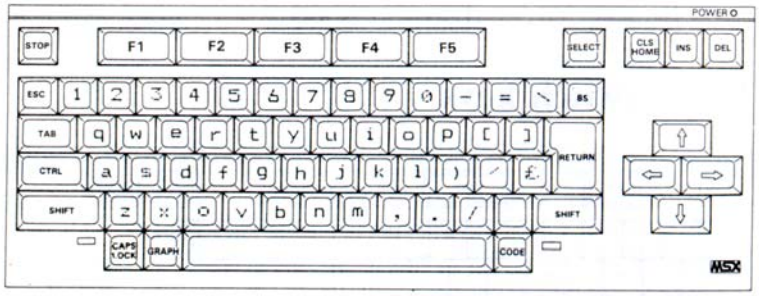


CAPS LOCK, SHIFT, CODE, and GRAPH keys are mode switch keys which are used to switch the character input mode. Even if the same key is pressed, different character is entered in different charactrer input modes.

(1) Normal mode

When the shaded keys shown in the figure below are pressed in the normal mode, symbols or lowercase letters are input as indicated in the figure.

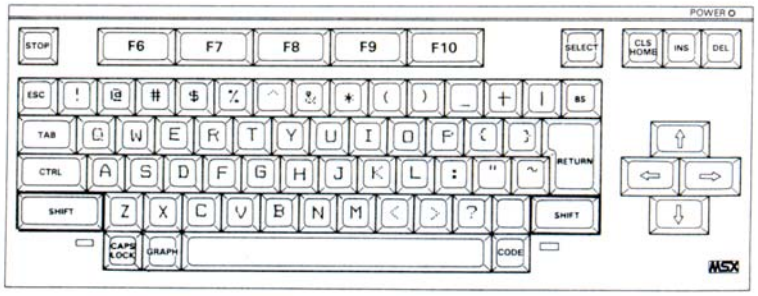
- **Normal mode**



Note:
The spacebar inputs a space regardless of what mode is selected.

When the shaded keys shown in the figure below are pressed together with the SHIFT key in the normal mode, symbols or uppercase letters are input as indicated in the figure.

- **Normal mode with SHIFT**



The [CAPS LOCK] key reverses the relationship of uppercase and lowercase letters with respect to the [SHIFT] key. (Only the letter keys are affected; [SHIFT] key operation is unchanged for other keys on the keybord.)

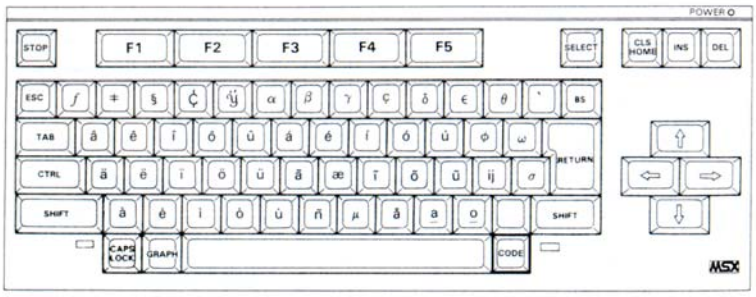
When the [CAPS LOCK] key is pressed once, the green lamp to the left of the key lights; when this lamp is lit, uppercase letters are input when the letter keys are pressed by themselves, and lowercase letters are input when they are pressed together with the [SHIFT] key. When the [CAPS LOCK] key is pressed a second time,the lamp goes out and the relationship between uppercase and lowercase letters is restored to it former state.

The [CAPS LOCK] key also function in this manner in the code mode and the graph mode.

(2) Code Mode

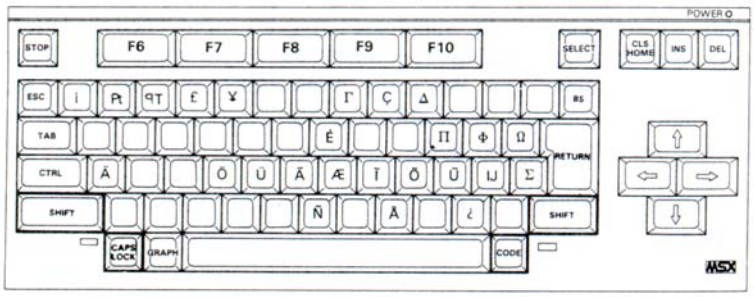
When the shaded keys shown in the figure below are pressed together with the [CODE] key, symbols or letters are input as indicated in the figure.

- **Code mode**



When the shaded keys shown in the figure below are pressed together with the [CODE] key and the [SHIFT] key, symbols or letters are input as indicated in the figure. Nothing is input by those keys in the figure which are blank.

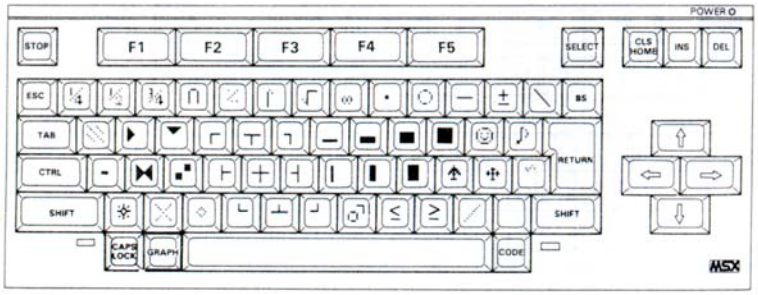
- **Code mode with [SHIFT]**



(3) Graph Mode

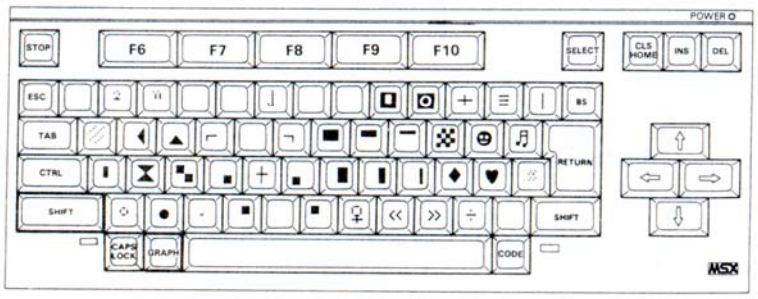
When the shaded keys shown in the figure below are pressed together with the [GRAPH] key, symbols or letters are input as indicated in the figure.

- **Graph mode**



When the shaded keys shown in the figure below are pressed together with both the [GRAPH] key and the [SHIFT] key, symbols or letters are input as indicated in the figure. Nothing is input by those keys in the figure which are blank.

- **Graph mode with [SHIFT] key**

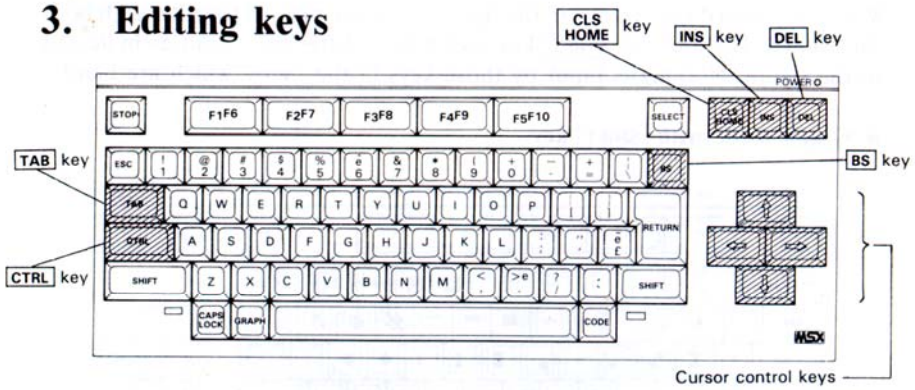


(4) Dead Key

The dead key (the [~ ^] key which is located above the [CODE] key) is used together with the [A], [E], [I], [O], [U], and [Y] keys to input the accented characters shown in the table below; this key has no effect on operation of other keys. To input the accented letters shown in the table, press the dead key, then press the applicable letter key in the mode indicated in the table.

Key operation	mode	Function	a	e	i	o	u	y	A	E	I	O	U	Y
	Normal	Grave	`	`	`	`	`							
+ [SHIFT]	Normal w/[SHIFT]	Acute	´	´	´	´	´			´				
+ [CODE]	Code	Circumflex	^	^	^	^	^							
+ [CODE] + [SHIFT]	Code w/Umlaut w/[SHIFT]		¨	¨	¨	¨	¨	¨	¨			¨	¨	
+ [GRAPH]	Graph	Grave	`	`	`	`	`							
+ [GRAPH] + [SHIFT]	Graph w/[SHIFT]	Acute	´	´	´	´	´			´				

3. Editing keys



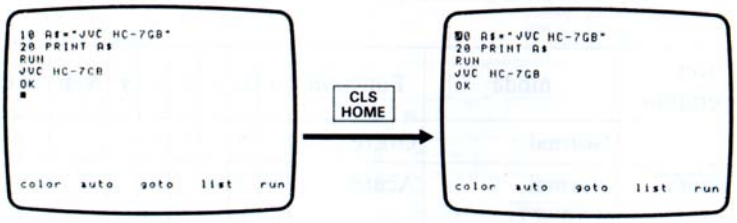
The editing keys are used to edit characters entered from the keyboard and control the screen.

- ⇧, ⇩, ⇨, ⇦ **(cursor control keys)**

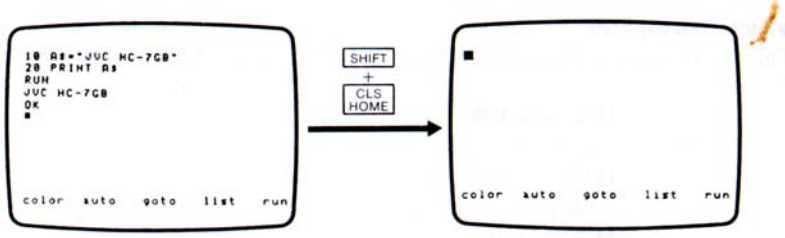
The cursor control keys move the cursor in the direction indicated by an arrow. When the cursor reaches the screen limit, no operation results even if the key is pressed.

- **CLS HOME** **(clear screen/home position)**

When this key is pressed, the cursor returns to the home position (the upper left corner of the screen).



Pressing the **CLS** key while holding down the **SHIFT** key clears the screen and moves the cursor to the home position. However, the function key display at the bottom is not cleared.



● **INS (insert)**

When this key is pressed, the size of the cursor is halved indicating that the insert mode is entered. The insert mode makes it possible to insert one or more characters at the left to the current cursor position. When the INS key or RETURN key is pressed during the insert mode, the insert mode is cancelled.

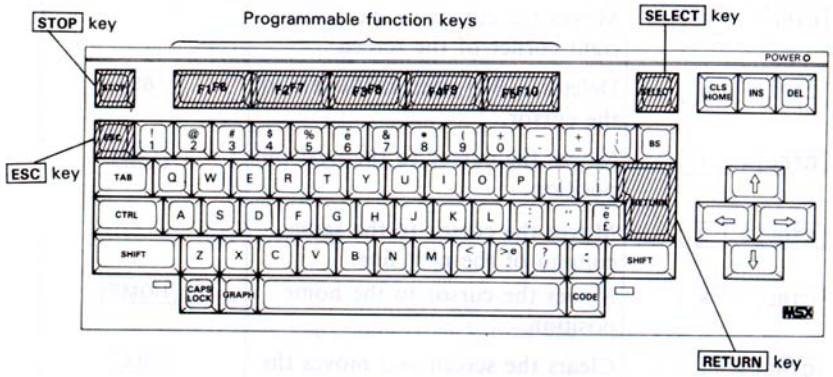
JVC C—7GB■
⇦
JVC C—7GB
INS
JVC C—7GB
H
JVC HC—7GB
⇨
JVC HC—7GB■ (□, ■ indicates the cursor.)

● **DEL (delete)**

This key is used to delete the character at the current cursor position.

HC—6 7GB■
⇦
HC—6 7GB
DEL
HC—7 7GB
⇨
HC—7GB■ (□, ■ indicate the cursor.)

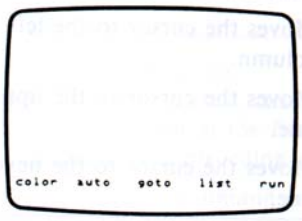
4. Function keys



A function key makes it possible to perform a predefined function or enter a command or statement with a single operation.

Note:
A command is a request which is entered directly from the keyboard to the computer and is executed as soon as the RETURN key is pressed. On the other hand, a statement is a part of the program, which is a command preceded by a line number.

- F1F6, F2F7, F3F8, F4F9, F5F10 (programmable function keys)
The character string assigned for each function key is displayed at the bottom line of the screen.



This assignment can be changed using the KEY statement. See "MSX BASIC GUIDE" for details on the KEY statement. The default assignment for each function key is shown in the table below.

F 1	F1 F6	color	F 6	SHIFT + F1 F6	color (15, 4, 4 ↙)
F 2	F2 F7	auto	F 7	SHIFT + F2 F7	cload (")
F 3	F3 F8	goto	F 8	SHIFT + F3 F8	cont (↙)
F 4	F4 F9	list	F 9	SHIFT + F4 F9	list. (↙)
F 5	F5 F10	run (↙)	F 10	SHIFT + F5 F10	(cls) run (↙)

Notes:

- Strings enclosed in parentheses are not displayed.
- SHIFT + F1F6 indicates that the F1F6 key is pressed while holding down the SHIFT key.
- A symbol " " indicates the return code (0DH).

- **STOP**

This key temporarily pauses execution of BASIC programs. To continue execution, press the STOP key again. Pressing CTRL and STOP together interrupts execution of BASIC programs and returns the BASIC interpreter to the command mode.

- **ESC (escape)**

This key is used in some application programs.

- **SELECT**

This key is used in some application programs.

- **RETURN**

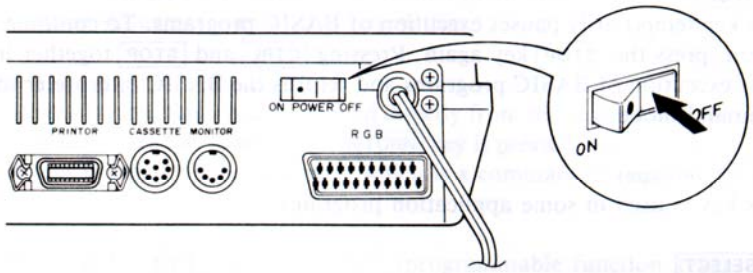
This key enters the return code (0DH). When this code is accepted by the computer, the command entered is executed. If an invalid command is entered, an error message is generated.

APPLICATION PROGRAMS

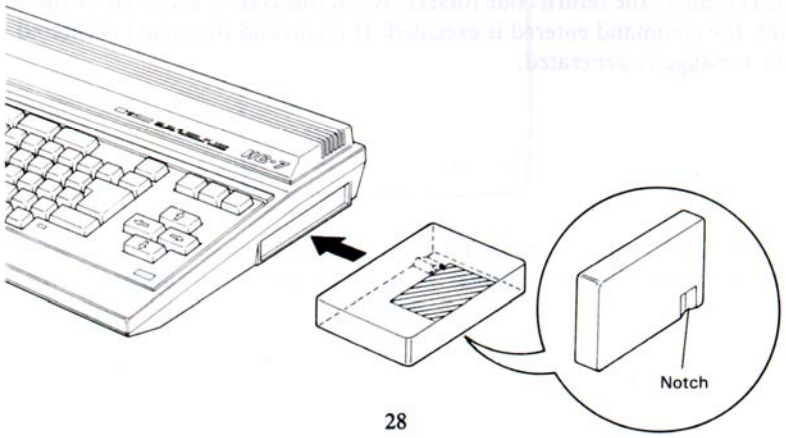
You can run all programs with the MSX mark available or programs developed with other MSX computers on the HC-7GB.
Application programs are provided in the form of ROM cartridge or cassette tape. This section explains how to use application programs in ROM cartridges and cassette tape.

1. Programs in ROM cartridges

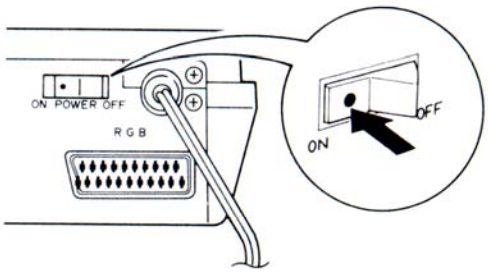
Follow the steps below to run the ROM cartridge program.
(1) Turn off the power of the HC-7GB.



(2) Insert the ROM cartridge into the slot at the rear or right side of the HC-7GB all the way in. In this case, insert the cartridge with the notch side facing down.



(3) Turn on the power of the HC-7GB.



(4) When the program in the ROM cartridge has been loaded into memory, the title screen appears. See the instruction for details on the screen and operation of the ROM cartridge.

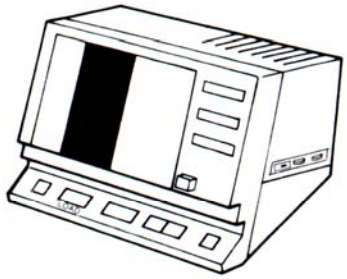
Note:
When cartridges are inserted in both slots, the side slot has the higher precedence.

2. Programs in cassette tape

① Operation differs according to the type of data recorder used.
② The explanations in this section are based on data recorder HC-R105.
③ For details, see your data recorder's instruction manual.

Data load

(1) Make sure that the data recorder is properly connected and the power switch is turned on.
(2) Load the cassette tape in the holder.
(3) Start the BASIC.
(4) Set "LOAD VOL" to 5 or higher and press the LOAD button.



Note:
If your data recorder does not have a REMOTE terminal, do steps (5) and (6) first.

(5) Enter the program name which must be enclosed in parentheses.

cload "TIMER"

Press keys as follows.

[C] [L] [O] [A] [D] ["] [T] [I] [M] [E] [R] ["]

Notes:
- Both uppercase and lowerecase letters can be used in commands. However, program names must be typed exactly the same as when programs were saved with the SAVE command.
- String "cload" can be entered by pressing the [F2 F7] key while holding down the [SHIFT] key.

(6) Press the [RETURN] key. The tape runs.
(7) When the program "TIMER" is found, the following message is displayed and the program is loaded into memory.

Found: TIMER

(8) When the program has been loaded, the BASIC prompt "OK" is displayed and the tape stops automatically Press the STOP button.
(9) Enter the RUN command or press the [F5] key to start the program.

Notes:

- When another program (e.g., TAPE) is found before program "TIMER" is found, the message

  Skip: TAPE

  is displayed and the tape runs until the specified program is found.
- When the program name is omitted in the CLOAD command, the program found first will be loaded.
- If an error occurs during load, reset the LOAD VOL control then try again.

Data save

(1) Make sure that the data recorder is properly connected and the power switch is turned on.
(2) Load the cassette tape into the holder.
(3) Press the SAVE and LOAD buttons at one time.

Note:
If your data recorder does not have a REMOTE terminal, do step (4) first.

(4) Enter the following command from the keyboard.

csave"JVC"

"JVC" is the name of the program to be saved. The program name consists of up to six characters and it begins with an alphabetical character.
(5) Press the [RETURN] key. The tape runs.
(6) When the data has been saved, BASIC prompt "OK" is displayed and the tape stops.

Verify

There are some cases when program cannot be properly saved on the cassette tape due to tape inferiority or noise. It is recommended that you verify the program saved on the cassette tape by the following procedures.

(1) Rewind the cassette tape on which the program has been saved.
(2) Press the LOAD button.

Note:
If your data recorder does not have a REMOTE terminal, do steps (3) and (4) first.

(3) Enter the following command. The name of the program to be verified is "JVC".

cload?"JVC"

(4) Press the RETURN key. The tape runs.
(5) When program "JVC" is found, the following message appears.

Found: JVC

(6) When the program has been verified, the tape stops automatically. When program has been checked and no error is detected, BASIC prompt "OK" is displayed; when an error is detected, the message below is displayed.

Verify error

OK

In this case, save the program on the tape again.

IN CASE OF DIFFICULTY

If you find an abnormality in your the HC-7GB while you are using it, take the checkup steps that follow. If the abnormality still persists, remove the power plug from the outlet and call your dealer for help.

HC-7GB will not power on.

- Check whether the power plug is securely connected to the outlet.

Nothing appears on the screen.

- Check whether the HC-7GB and monitor display power switches are on.
- Check cables.
- Check whether the ROM cartrige is properly inserted.
- Check whether channels are closely turned.

ROM cartridge will not function.

- Check whether the ROM cartridge is snugly inserted.
- Check for dirty cartridge terminals.

Program cannot be loaded from data recorder.

- Check for proper connection between the computer and data recorder.
- Check whether the data recorder volume control setting is too low (set to scale 5 or higher).
- Check for proper phase (if phase switchover control is provided).

Computer stopped during execution.

- Check for power failure.
- Check whether the POWER switch is in the OFF position.
- Check whether the power plug is disconnected from the power outlet.
- Check whether peripherals are properly connected.
- Check for the presence of electrical disturbances from other electrical appliances. If there is a possibility of electrical disturbances, use a power outlet immune to such interferences, use a power line filter dedicated for personal computers.

HARDWARE SPECIFICATIONS

- **CPU and memory:** Z80A compatible microprocessor
  Clock frequency: 3.58 MHz
  64K RAM
  32K ROM (MSX BASIC)
- **Keybord:** UK version
- **Display:**

Character set:	Uppercase alphanumeric characters, graphics symbols (252 symbols, 8 × 8 dot matrix)
Colors:	16 colors (including black and transparent)
Screen size:	Screen mode 1: 32 characters × 24 lines (Initially set to 29 characters × 24 lines) Screen mode Ø: 40 characters × 24 lines (Initially set to 37 characters × 24 lines)
Graphics:	256 × 192 dots
Sprites:	32

Note:
If clipping occurs due to the mismatch in the holizontial width between the computer and the TV set, reset the line width with the WIDTH BASIC statement.

- **Sound function:** 8 octaves, 3 parts, 1 noise
- **Power and miscellaneous specifications:**

Source voltage:	240 V AC ± 10%
Frequency:	50 Hz
Power consumption:	12 W
Environmental conditions:	Temperature: 0 to 35°C Humidity: 80 %RH or maximum

- **Physical specifications:**

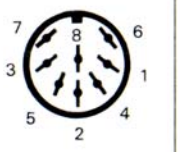
Dimensions:	382 mm (W) × 86 mm (H) × 286 mm (D)
Weight:	2.7 kg

* Specifications and appearance are subject to change without notice.

INTERFACES

- **Cassette interface:**

FSK encoding
1,200 bauds Default
2,400 bauds Software programmable
DIN 8-pin (female) connector

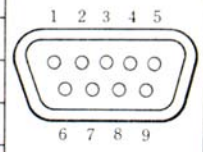
Pin No.	Signal name	Pin assignment
1	GND	
2	GND	
3	GND	
4	CMTOUT	
5	CMTIN	
6	REM+	
7	REM−	
8	GND	

- **A/V interface:**

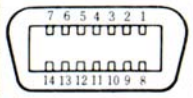
PAL system
DIN 5-pin (female) connector

Pin No.	Signal name	Pin assignment
1	NC	
2	GND	
3	SOUND	
4	VIDEO	
5	NC	

● **General-purpose interface:** D type connector (male) × 2, TTL level

Pin No.	Signal name	Pin assignment
1	FWD	
2	BACK	
3	LEFT	
4	RIGHT	
5	+5	
6	TRG1	
7	TRG2	
8	OUTPUT	
9	GND	

● **Printer interface:** 8-bit parallel Centronics type interface
14-pin TTL-level female connector

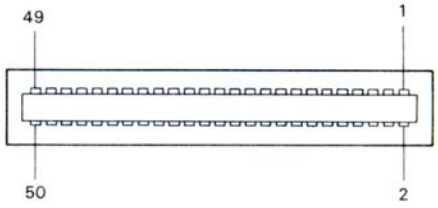
Pin No.	Signal name	Pin assignment
1	$\overline{\text{PSTB}}$	
2	PDB0	
3	PDB1	
4	PDB2	
5	PDB3	
6	PDB4	
7	PDB5	
8	PDB6	
9	PDB7	
10	NC	
11	BUSY	
12	NC	
13	NC	
14	GND	

Caution:
Do not use any commercially available printer interface cartridge.

● **Cartridge bus:** 50-pin MSX-compatible female connector

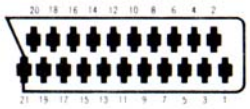
No.	Name	I/O	No.	Name	I/O	No.	Name	I/O
1	$\overline{CS1}$	O	2	$\overline{CS2}$	O	3	$\overline{CS12}$	O
4	$\overline{SLTSL}$	O	5	Reserved*	—	6	$\overline{RFSH}$	O
7	$\overline{WAIT}$	I	8	$\overline{INT}$	I	9	$\overline{M1}$	O
10	$\overline{BUSDIR}$	I	11	$\overline{IORQ}$	O	12	$\overline{MERQ}$	O
13	$\overline{WR}$	O	14	$\overline{RD}$	O	15	$\overline{RESET}$	O
16	Reserved*	—	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	—			

I/O: Indicates the signal flow with respect to the computer.
* : Reserved pins are not accessible to the user.



- **RGB connector:**

Pin No.	Signal name
1	Audio out (500mVrms/1kΩ)
2	NC
3	Audio out (500mVrms/1kΩ)
4	Audio GND
5	Blue GND
6	NC
7	Blue out (0.7Vp-p/75Ω)
8	Status CVBS (''H'' pull-up)
9	Green GND
10	NC
11	Green out (0.7Vp-p/75Ω)
12	NC
13	Red GND
14	NC
15	Red out (0.7Vp-p/75Ω)
16	Status RGB (''H'' pull-up)
17	CVBS GND
18	Status RGB GND
19	Sync. out (0.5Vp-p/75Ω, Negative Sync.)
20	NC
21	Socket GND



- **RF out connector:** PAL CCIR I 36 channel
  RCA type Pinjack

BASIC COMMANDS AND STATEMENTS

The MSX-BASIC Version 1.0 commands are statements are listed below.

AUTO

Format: [AUTO <start line number>[,<increment>]]
Function: Automatically generates program line numbers.

CONT

Format: CONT
Function: Resumes program execution after a break.

DELETE

Format: DELETE [<start line number>] [– <end line number>]
Function: Deletes part of program lines.

LIST/LLIST

Format: LIST [<start line number>] [– <end line number>]
LLIST [<start line number>] [– <end line number>]
Function: Lists a program on the screen or a printer.

NEW

Format: NEW
Function: Deletes the program currently in memory and clears all variables.

RENUM

Format: RENUM [<new line number>][,[<old line number>] [,<increment>]]
Function: Renumbers program lines.

RUN

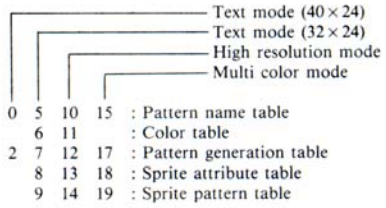
Format: RUN [<line number>]
Function: Starts execution of a program.

SAVE/LOAD/MERGE

Format: SAVE <device descriptor>
LOAD <device descriptor>
MERGE <device descriptor>
Function: Saves a program on or loads or merges a program from the specified device. All data is saved, loaded, or merged in the ASCII format.

BASE (system variable)

Format: BASE (<integer expression>)
Function: Writes and reads a base address to and from the VDP table. Users who do not understand the VDP standards must not use this command. <integer expression> takes the following values:



BSAVE

Format: BSAVE <device descriptor>, <start address>, <end address> [,<execution start address>]
Function: Saves machine language data on a specified device. BASIC assumes the start address as the execution start address when no execution start address is given.

BLOAD

Format: BLOAD <device descriptor>[,R][,<offset>]
Function: Loads a machine language program from a specified device. When the R option is specified, BASIC starts execution at the address specified in BSAVE after loading the program.

CSAVE

Format: CSAVE "<filename>"[,<baud rate option>]
Function: Saves a program on cassette tape. The baud rate option specifies the baud rate value 1 (1200 baud) or 2 (2400 baud).

CLOAD

Format: CLOAD ["<filename>"]
Function: Loads a program from tape. The baud rate is identified automatically.

CLOAD?

Format: CLOAD? ["<filename>"]
Function: Compares the program on tape with that in memory.

TRON/TROFF

Format: TRON/TROFF
Function: Starts and stops tracing program execution.

General statements

CLEAR

Format: CLEAR [<string area size>[,<maximum memory address>]
Function: Initializes all variables and defines sizes of the string and user areas.

CALL

Format: CALL <extended statement name> [(<argument list>)]
Function: Executes an extended statement. "_" may serve as "CALL".

CLOSE

Format: CLOSE <file number>
Function: Closes the specified file.

DATA

Format: DATA <constant list>
Function: Stores numeric or string constants read by READ statements.

DIM

Format: DIM <variables with subscripts>
Function: Declares the maximum value for array variable subscripts.

DEFINT/DEFSNG/DEFDBL/DEFSTR

Format: DEFINT <variable name range>
Function: Declares variable types as integer, single precision, double precision, or string.
Format: DEFSNG <variable name range>
Function: Declares variable types as integer, single precision, double precision, or string.
Format: DEFDBL <variable name range>
Function: Declares variable types as integer, single precision, double precision, or string.
Format: DEFSTR <variable name range>
Function: Declares variable types as integer, single precision, double precision, or string.

DEFFN

Format: DEFFN <function name> [[<argument list>]] = <definition expression>
Function: Defines a user-defined function.

DEF USR

Format: DEF USR [<digit>] = <expression>
Function: Defines the start address of a machine language subroutine. <digit> may be a number 0 to 9 and defaults to 0.

ERASE

Format: ERASE <array name list>
Function: Deletes an array from a program.

END

Format: END
Function: Terminates the execution of a program.

ERROR

Format: ERROR <integer expression>
where <integer expression> is an error code.
Function: Simulates the occurrence of an error.

FOR ~ NEXT

Format: FOR <variable name> = <initial value> TO <final value> [STEP<increment>]
NEXT [<variable name>[,<variable name>...]]
Function: Executes a series of instructions between FOR and NEXT a given number of times.

GOSUB ~ RETURN

Format: GOSUB <line number>
RETURN
Function: Invokes and returns from a subroutine.

GOTO

Format: GOTO <line number>
Function: Branches to a specified line.

IF ~ THEN/IF ~ GOTO

Format: IF <expression> THEN <statement>/ <line number>
IF <expression> GOTO <line number>
Function: Makes a decision based on the result of the expression.

IF ~ THEN ~ ELSE

Format: IF <expression> THEN <statement> / <line number>
ELSE <statement> / <line number>
Function: Makes a decision based on the result of the expression.

IF ~ GOTO ~ ELSE

Format: IF <expression> GOTO <line number>
ELSE <statement> / <line number>
Function: Makes a decision based on the result of the expression.

INPUT

Format: INPUT ["<prompt string>";] <variable list>
Function: Assigns values to the specified variables.

KEY

Format: KEY (<expression>),<string expression> where <expression> is a PH key number.
Function: Sets a programmable function key to a string.

KEY LIST

Format: KEY LIST
Function: Displays the programmable function key values.

LINE INPUT

Format: ["<prompt string>";] <character variable list>
Function: Enters strings on a line basis.

LET

Format: [LET] <variable name> = <expression>
Function: Assigns a value to a variable (LET may be omitted).

MAXFILES

Format: MAXFILES = <expression>
Function: Defines the maximum number of files that may be open at any one time. The value of the expression may be a value 0 to 15. Only LOAD and SAVE are available when 0 is specified.

MID $

Format: MID $ (<string 1>, expression 1 [,expression 2]) = <string 2>
Function: Replaces a string of length expression 2 from string 1 beginning with position given by expression 1 with expression 2.

ON ERROR GOTO

Format: ON ERROR GOTO <line number>
Function: Specifies the starting line number of an error processing routine.

ON GOTO

Format: ON <expression> GOTO <line number list>
Function: Branches to one of the specified lines depending on the value of the expression.

GOSUB

Format: ON <expression> GOSUB <line number list>
Function: Branches to one of the specified lines depending on the value of the expression.

~ ON/OFF/STOP

Format: KEY <expression> <ON/OFF/STOP>
Function: Enables or disables function key generated interrupts.

Format: STRIG [(<expression>)] <ON/OFF/STOP>
Function: Enables or disables interrupts generated by a joystick trigger.

Format: STOP <ON/OFF/STOP>
Function: Enables or disables interrupts generated by CTRL + STOP.

Format: SPRITE <ON/OFF/STOP>
Function: Enables or disables interrupts generated on a sprite pattern superimposition.

Format: INTERVAL <ON/OFF/STOP>
Function: Enables or disables interval timer interrupts.

ON ~ GOSUB

Format: ON KEY GOSUB <line number list>
Function: Specifies the starting line number of the interrupt routine that is to be invoked by a function key. The line number list may consist of up to 10 line numbers which correspond to the individual function keys.

Format: ON STOP GOSUB <line number>
Function: Defines the starting line number of the interrupt that is to be routine invoked by the STOP key.

Format: ON STRING GOSUB <line number list>
Function: Defines the starting line number of the interrupt routine that is to be invoked by the joystick trigger button.

Format: ON SPRITE GOSUB <line number>
Function: The line number list may consist of up to 5 line numbers which correspond to the trigger buttons on the two joy sticks and the space key.

Format: ON INTERVAL = <interval time>
Function: Defines the starting line number of the interrupt routine that is to be invoked on a sprite pattern superimposition.

Format: GOSUB <line number>
Function: Defines the starting line number of the interrupt routine that is to be invoked by the interval timer.

OPEN

Format: OPEN "<device descriptor>[<file name>]"
[FOR <INPUT/OUTPUT/APPEND>]
AS [#] <file number>
Function: Opens a device and defines its I/O mode. The device descriptor must be one of the following:
CAS: Cassette tape
CRT: CRT display
GRP: Graphic screen (when displaying characters on the graphic screen.)
LPT: Printer

OUT

Format: OUT <integer expression 1>, <integer expression 2>
Function: Transfers the value of integer expression 2 to the port at the address specified in <integer expression 1>.

POKE

Format: POKE <memory address>, <integer expression>
Function: Writes the value of an integer expression into the specified memory address.

PRINT

Format: PRINT <variable list>
Function: Outputs the contents of variables to the screen/printer.

LPRINT

Format: LPRINT <variable list>
Function: Outputs the contents of variables to the screen/printer.

PRINT USING

Format: PRINT USING <format control character string>;<variable list>
Function: Outputs the contents of variables in the specified format to the screen/printer.

LPRINT USING

Format: LPRINT USING <format control character string>;<variable list>
Function: Outputs the contents of variables in the specified format to the screen/printer.

PRINT

Format: PRINT #<file number>, <variable list>
Function: Writes data to or reads data from the specified file.

INPUT

Format: INPUT #<file number>, <variable list>
Function: Writes data to or reads data from the specified file.

READ

Format: READ <variable list>
Function: Reads the constants specified in DATA statements into variables.

REM

Format: REM <character string>
Function: Inserts explanatory remarks into programs.

RESTORE

Format: RESTORE <line number>
Function: Specifies the starting line number of the DATA statements to be read by the READ statement.

RESUME

Format: RESUME
Function: Resumes program execution at the specified line number after the termination of error processing.
Format: RESUME 0
Function: Resumes program execution at the specified line number after the termination of error processing.
Format: RESUME NEXT
Function: Resumes program execution at the specified line number after the termination of error processing.
Format: RESUME <line number>
Function: Resumes program execution at the specified line number after the termination of error processing.

STOP

Format: STOP
Function: Terminates program execution and returns control to the command level.

SWAP

Format: SWAP <variable name 1>, <variable name 2>
Function: Swaps the contents of variables 1 and 2. The type of the two variables must match.

TIME

Format: TIME = <expression>
Function: Sets the internal clock. <expression>may be a number 0 to 65535 can be counted. The internal clock is incremented by 1 every 1/60 second.

WAIT

Format: WAIT <port number>, I [,J]
Function: Suspends program execution and monitors the status of a machine I/O port.

VDP (system variable)

Format: VDP [<number>]
Function: Read or writes the contents of the VDP register specified by <number>. <number>must be an integer from 0 to 8. 8 is for reading only. Users who have little knowledge about the VDP standards must not use this statement.

Graphics and Sound Statements

BEEP

Format: BEEP
Function: Sounds the internal buzzer.

CIRCLE

Format: CIRCLE(<X-coordinate>,<Y-coordinate>),<radius>[,<color>]
[,<start point>][,<end point>][,<ratio>]
Function: Draws a circle as specified by the parameters.

CLS

Format: CLS
Function: Clears the screen.

COLOR

Format: COLOR[<foreground>>][,<background>][,<border>>]
Function: Specifies the color of the foreground, background, and border. The default values are 15, 4, 4, respectively. The color codes are:

0	Transparent	8	Red
1	Black	9	Light red
2	Green	10	Dark yellow
3	Light green	11	Light yellow
4	Dark blue	12	Dark green
5	Blue	13	Violet
6	Dark red	14	Grey
7	Cyan	15	White

DRAW

Format: DRAW <string expression>
Function: Draws a graphics pattern according to the graphics macro language specified by <string expression>

LINE

Format: LINE(<X-coordinate>,<Y-coordinate>[-(<X-coordinate>,<Y-coordinate>)]
[,<color>][,B/BF]
Function: Draws a line or rectangle in the specified color.

LOCATE

Format: LOCATE [<X-coordinate>,<Y-coordinate>][,<cursor switch>]
Function: Positions the cursor in the specified coordinates.

PUT SPRITE

Format: PUT SPRITE <sprite face number>[,(<X-coordinate>,<Y-coordinate>)]
[,<color code>][<,sprite number>]
Function: Displays the specified sprite pattern. X-coordinate is a number from −32 to 255 and Y-coordinate is a number from −32 to 191. The specified sprite is deleted when Y-coordinate is set to 209 and the sprites after the specified face are deleted when Y-coordinate is set to 208.

PAINT

Format: PAINT(<X-coordinate>,<Y-coordinate>)[,<color>] [,<border color>]
Function: Paints the specified area.

PLAY

Format: PLAY <string>,<string>,<sting>
Function: Generates a sound as specified by the music macro language.

PSET

Format: PSET (<X-coordinate>,<Y-coordinate>)[,<color>]
Function: Draws a dot or deletes a dot in the specified coordinates.

PRESET

Format: PRESET (<X-coordinate>,<Y-coordinate>)
Function: Draws a dot or deletes a dot in the specified coordinates.

SCREEN

Format: SCREEN <mode>[,<sprite size>]
[,<key click switch>]
[,<cassette baud rate>]
[,<printer options>]

Function: Sets the screen mode and cassette baud rate.

The legitimate screen modes are:

0 : Text mode (40 × 24)
1 : Text mode (32 × 24)
2 : High resolution graphics mode
3 : Multi-color mode

The legitimate sprite sizes are:

0 : 8 × 8 pixels (standard)
1 : 8 × 8 pixels (enlarged)
2 : 16 × 16 pixes (standard)
3 : 16 × 16 pixes (enlarged)

A key click switch value of 1 turns on the key click function and a value of 0 turns off the function. The cassette baud rate value of 1 corresponds to 1200 bauds and 2 to 2400 bauds. The printer option specifies whether a printer having the MSX character set is to be used. A nonzero value specifies that any graphics characters are to be converted into blanks.

SPRITE $

Format: SPRITE $(<expression>)=(<string>)
Function: Defines a sprite pattern.
The possible expression values and string length are:
For SCREEN option value of
0 or 1 : 256 maximum for expression and 8 characters for string length
2 or 3 : 64 maximum for expression and 32 characters for string length

VPOKE

Format: VPOKE <VRAM address>,<integer expression>
Function: Writes the value of the integer expression directly into the specified VRAM address. The VRAM address must be a number from 0 to 16383 and the integer expression must evaluates to a number between 0 and 255.

KEY

Format: KEY <ON/OFF>
Function: Turns on or off the display of function key definitions on the bottom of the screen.

MOTOR

Format: MOTOR [<ON/OFF>]
Function: Turns on or off the cassette recorder motor.

SOUND

Format: SOUND <ON/OFF>
Function: Specifies whether the PSG output is to be directed to the cassette output.

SOUND

Format: SOUND <expression 1>,<expression 2>
Function: Writes the value of <expression 1> into the PSG register designated by <expression 2>.

STEP

Format: STEP (<X-coordinate>,<Y-coordinate>)
Function: Gives the relative position with respect to the last established coordinates. This statement allows relative coordinates to be used in LINE, PAINT, and other graphics statements.

WIDTH

Format: WIDTH <expression>
Function: Defines the width of the screen. <expression> may take a value of 1 to 32 or 40.

Functions

BIN $

Format: BIN$(<expression>)
Function: Converts a binary number in <expression> into a character string representing the number.

POINT

Format: POINT(<X-coordinate>,<Y-coordinate>)
Function: Returns the color of the specified coordinates.

VPEEK

Format: VPEEK (<VRAM address>)
Function: Returns the contents of the specified VRAM address.
<VRAM address> value must be between 0 and 16383.

STICK

Format: STICK (<expression>)
Function: Returns the orientation of the joystick identified by <expression>.
<expression> takes a value from 0 to 2. A 0 specified in <expression> identifies the cursor key. 1 specifies port1, and 2 identifies the joystick connected to port 2.

STRING

Format: STRIG(<expression>)
Function: Returns the status of the trigger button of the joystick identified by <expression> <expression> takes a value from 0 to 4. A 0 specified in <expression> identifies the space key, 1 or 3 identifies port 1, and 2 or 4 identifies the joystick connected to port 2. STRIG returns −1 if the trigger button is pressed and 0 otherwise

PDL

Format: PDL (<expression>)
Function: Returns the status of the paddle identified by <expression>. <expression> takes a value from 1 to 12. An odd number specified in <expression> identifies port 1 and an even number identifies port 2.

PAD

Format: PAD (<expression>)
Function: Returns the status of the touch pad identified by <expression>. An <expression> value of 0 to 3 identifies port 1 and a value of 4 to 7 identifies port 2. When the button on the touch pad is pressed, PAD returns the X-coordinate at which the button is pressed if the value of <expression> is 1 or 5 and the Y-coordinate if the <expression> value is 2 or 6. When the <expression> value is 3 or 7, PAD returns −1 if the button on the the specified touch pad is pressed and 0 otherwise.

PLAY

Format: PLAY(<expression>)
Function: Returns −1 if the channel identified by <expression> is executing the PLAY statement and 0 otherwise. <expression> can takes on a value from 0 to 3. A 0 in <expression> indicate all channels and a number from 1 through 3 indicate the corresponding channel.

TIME

Format: TIME
Function: Returns the current value of the internal clock.

Arithmetic, String and Special Functions

ABS

Format: ABS(numeric expression)
Function: Returns the absolute value of the argument.

ASC

Format: ASC(string expression)
Function: Returns the ASCII code for the string expression.

ATN

Format: ATN(numeric expression)
Function: Calculates the arctangent (inverse tangent) of the argument.

CDBL

Format: CDBL(<numeric expression>)
Function: Converts the value of numeric expression to a double precision number.

CHR$

Format: CHR$(numeric expression)
Function: Returns the character equivalent for an ASCII code.

CINT

Format: CINT(<numeric expression>)
Function: Converts the value of a numeric expression to an integer.

COS

Format: COS(numeric expression)
Function: Calculates the cosine of the argument.

CSNG

Format: CSNG(<numeric expression>)
Function: Converts the value of numeric expression to a single precision number.

CSRLIN

Format: CSRLIN
Function: Returns the line number on which the cursor rests.

EXP

Format: EXP(numeric expression)
Function: Calculates e to the power of the argument.

EOF

Format: EOF(<file number>)
Function: Returns −1 if end of a sequential file has been reached, and returns 0 otherwise.

ERR/ERL

Format: ERR/ERL
Function: Returns the error code and line number associated with an error.

FRE(0)/FRE(" ")

Format: FRE(0)/FRE(" ")
Function: Returns the number of bytes in memory not being used. FREE returns the number of free bytes in the text area or the string space as specified by the argument.

FIX

Format: FIX(<numeric expression>)
Function: Truncates the numeric expression to an integer.

HEX$

Format: HEX$(<numeric expression>)
Function: Converts an integer given by the numeric expression to a hexadecimal value and returns a string representing the hexadecimal value.

INKEY$

Format: INKEY$
Function: Reads a character from the keyboard if any key is pressed. Otherwise, returns a null string.

INPUT$

Format: INPUT$(<numeric expression>)
Function: Reads data items of specified length from the specified device.

INSTR

Format: INSTR([<numeric expression>,]<string 1>,<string 2>)
Function: Searches for string 2 in string 1 and returns the position at which a match is found.

INT

Format: INT(numeric expression)
Function: Returns the largest integer which is less than or equal to the numeric expression.

LEFT$

Format: LEFT$(<string>,<numeric expression>)
Function: Returns the leftmost characters of specified length from the string.

LEN

Format: LEN(<string>)
Function: Returns the number of characters in the string.

LOG

Format: LOG(numeric expression)
Function: Calculates the natural logarithm of the argument.

LPOS

Format: LPOS(numeric expression)
Function: Returns the current position of the print head within the printer buffer.

MID$

Format: MID$(<string>,<numeric expression 1>[,<numeric expression 2>])
Function: Returns the requested part of a given string.

OCT$

Format: OCT$(<numeric expression>)
Function: Returns a string which represents the octal value of the integer value of the numeric expression.

PEEK

Format: PEEK(numeric expression)
Function: Returns the byte read from the indicated memory position.

POS

Format: POS(numeric expression)
Function: Returns the column position of the cursor on the screen.

RIGHT$

Format: RIGHT$(<string>,<numeric expression>)
Function: Returns the rightmost characters of specified length from the string.

RND

Format: RND(numeric expression)
Function: Returns a random number.

SGN

Format: SGN(numeric expression)
Function: Returns a number representing the algebraic sign of the argument.

SIN

Format: SIN(numeric expression)
Function: Calculates the sign of the argument.

SPACE$

Format: SPACE$(numeric expression)
Function: Returns a string consisting of spaces of specified length.

SPC

Format: SPC(numeric expression)
Function: Prints the spaces of the length indicated by the value of the numeric expression.

SQR

Format: SQR(numeric expression)
Function: Returns the square root of the argument. Used only in PRINT and LPRINT.

STR$

Format: STR$(numeric expression)
Function: Returns a string representation of the value of numeric expression.

STRING$

Format: STRING$(<numeric expression 1>,<numeric expression 2>/<string>)
Function: Returns a requested string of requested length.

TAB

Format: TAB(numeric expression)
Function: Prints spaces at the a position on the line on which the cursor rests.

TAN

Format: TAN(numeric expression)
Function: Calculates the tangent of the argument.

USR

Format: USR [<numeric value>](<expression>)
Function: Calls the indicated machine language subroutine. <numeric value> is one of the digits 0 through 9.

VAL

Format: VAL(<string>)
Function: Returns the numerical value of the string.

VARPTR

Format: VARPTR(<variable name>/#<file number>)
Function: Returns the address in memory that store a variable or file variable.

ERROR MESSAGES

Bad file name **Error code 56**

An invalid file structure was found.

- An attempt was made to open a file with an invalid file name.
- An attempt was made to read a file which was opened for output, or vice versa.

Bad file number **52**

An invalid file number was found.

- A PRINT # statement was executed specifying a file number for a file which was not opened file name.
- A file number 2-15 was used without defining it in the MAXFILES statement.

Can't CONTINUE **17**

The program cannot resume execution.

- The execution was interrupted by an error.
- The program has been modified after the interruption.

Device I/O error **19**

An I/O error occurred in a peripheral device.

- Flaws were found on a cassette tape.
- Poorly adjusted cassette recorder output level.
- The command was interrupted because power to the printer or cassette recorder was shut down.

Direct statement in file **57**

An invalid statement was found in an ASCII format program file.

- A statement with no line number was found in the program file.
- The file being loaded contains text other than BASIC programs.

Division by zero **11**

An attempt was made to divide by 0.

- The divisor is not 0.
- An attempt was made to divide a quantity by an undefined variable.

FIELD overflow **50**

Error condition reserved for future expansion.

- An area larger than 256 bytes was specified in a FIELD statement.

File already open **54**

The requested file was already open.

- An OPEN statement with a file number of the existing file was executed when the file was not closed.

File not OPEN **59**

Open processing failed.

- The file number specified in a PRINT # or INPUT # statement was not yet defined with an OPEN statement.

File not found **53**

The specified file does not exist. Error condition reserved for future expansion.

- An invalid file name was specified.
- An invalid file extension was specified.

Out of string space **Error code 14**

No adequate character string area.

- The string area to be specified in the CLEAR statement is too small.

Overflow **6**

A numeric value exceeded the allowed range.

- The result of an integer calculation is smaller than −32768 or greater than 32767.
- The result of a real number calculation is smaller than −9.99...99E62 or greater than 9.99...99E62.

14 9s 14 9s

- The address specified as a parameter exceeds the allowed range.

RESUME without error **22**

A RESUME statement was executed when no error had occurred.

- A branch was made to an error routine from a GOTO or GOSUB statement.
- The main routine contains no END statement and the error routine immediately following the main routine was executed.

RETURN without GOSUB **3**

A RETURN statement was encountered before a GOSUB statement.

- Control was transferred from a GOTO statement to a subroutine.
- The main routine contains no END statement and the error routine immediately following the main routine was executed.

Redimensioned array **10**

An array was defined in duplicate.

- An array with a duplicate name was defined without executing an ERASE statement.
- An array variable was used without declaration before a DIM statement was executed for the variable.
- A DIM statement was found in a loop.

Sequential I/O only **58**

Error condition reserved for future expansion.

- The current BASIC allows file input/output only for sequential files.

String formula too complex **16**

A too complex string expression was encountered.

- The string expression specified on a single line is too complex or the nesting of parentheses is too deep.

String too long **15**

A too long string was found.

- An attempt was made to assign a string longer than 256 characters to a string variable.

Subscript out of range **9**

A subscript of an array exceeds the defined range.

- A subscript value is too large.
- The subscript value of an undefined array is larger than 11.

Illegal direct **Error code 12**

An attempt was made to execute a statement not allowed in the direct mode.

- The DEFFN statement is not allowed in the direct mode.

Illegal function call **5**

An invalid statement or function reference was encountered.

- A parameter specifying an invalid value was found in a statement or function.
- A negative subscript was found in an array.

Input past end **55**

An attempt was made to read past the end of a file.

- The number of executions of the INPUT # or LINEINPUT # statement is larger than the number of data items in the file.

Internal error **51**

An error was found in the interpreter.

- This error may occur if the POKE statement or USR function is used improperly.

Line buffer overflow **25**

The input line buffer becomes fill.

- The line buffer can contain a maximum of 255 lines.

Missing operand **24**

A required parameter was not found.

- A wrong number of parameters was specified.
- Numbers are separated by not commas but period.

NEXT without FOR **1**

A FOR statement is missing.

- FOR loops nested improperly.
- Control is transferred directly inside of a FOR ~ NEXT loop.

No RESUME **21**

No RESUME statement was specified in an error routine.

- An error routine must be terminated by either END, RESUME, or ON ERROR GOTO statement.
- A GOTO statement was used to return from an error routine.

Out of DATA **4**

No data to be read with the READ statement.

- An inadequate number of data items were specified.
- The delimiter symbols was improperly used in a DATA statement.

Out of memory **7**

Main memory is inadequate.

- The program is too long.
- Too many variables were used.
- Too large an array was used.
- The nesting level of FOR ~ NEXT or GOSUB ~ RETURN statements is too deep.
- Too many space was reserved with a CLEAR statement.

Syntax error **Error code 2**

A statement which did not comply with the MSX BASIC grammar was encountered.

- An item other than keywords (reserved words) was found.
- Parentheses are not balanced.
- An invalid delimiter (comma, period, colon, or semicolon) was found.
- A variable name not beginning with an alphabetic character was found.
- A variable containing a keyword (reserved word) was found.
- An invalid parameter was found in a function reference or statement.
- Invalid input data.

Type mismatch **13**

The types of variables do not match.

- An attempt was made to assign a character constant to a numeric variable.
- An attempt was made to assign a numeric constant to a character variable.
- The types of function parameters do not match.

Undefined line number **8**

An undefined line number was encountered.

- A statement was specified with no line number.
- The line number specified in the GOTO, GOSUB, RESTORE, or RUN statement does not exist.

Undefine user function **18**

The requested user function is not defined.

- A variable beginning with FN was found.
- An invalid function name was specified in a DEFFN statement.
- No DEFFN statement was executed (execution was started in the middle of the program with a GOTO statement).

Unprintable error **23, 26 to 49, 60 to 255**

An error condition which could not be identified by error codes occurred.

- An ERROR statement was executed.

Verify error **20**

An error was found while verifying a program file on a cassette tape.

- A program was verified with a CLOAD? command but a mismatch occurred between the programs on the tape and in the text area.

JVC

VICTOR COMPANY OF JAPAN, LIMITED

Printed in Japan
HC-7GB-IB-A
Ⓐ1