

YAMAHA

YRM-303

MIDI MACRO & MONITOR

OWNER'S MANUAL

MANUEL D'UTILISATION

BEDIENUNGSANLEITUNG

CHAPTER I

GETTING STARTED

THE COMPONENTS OF YOUR SYSTEM

Here is a list of the components you need to enjoy the full potential of the MIDI Macro & Monitor program.

Computer & Peripherals

- **Yamaha Music Computer or Yamaha MSX computer equipped with a MIDI interface** The Yamaha Music Computers (CX5M or CX5MII) come with the SFG-01 or the SFG-05 Yamaha FM Sound Synthesizer Unit. If you own another Yamaha MSX computer you will have to purchase a MIDI interface (see below).
- **MIDI Interface** You can use the Yamaha FM Sound Synthesizer Units SFG-01 or SFG-05 or the SMD-01 as a MIDI interface for your computer.
- **Color Monitor or TV** We recommend an RGB monitor for optimal screen resolution. If your computer is not equipped with an RGB connector, any color monitor or TV will do. Please refer to the Owner's Manual of your computer for more details.
- **Printer** A printer will provide you with hard copies of your programs. Any printer bearing the **MSX** label can be used. We recommend the Yamaha PN-101 Dot Impact Printer.
- **External memory** You can use a Data Recorder (or an ordinary tape recorder). For high-speed memory operation, however, a Yamaha Memory Cartridge (UDC-1) or an MSX Floppydisk Drive (the Yamaha FD-05, for example) is best. Note that only one floppy disk drive can be used and that only the SFG-05 is compatible with a floppy disk drive.

MIDI Peripherals

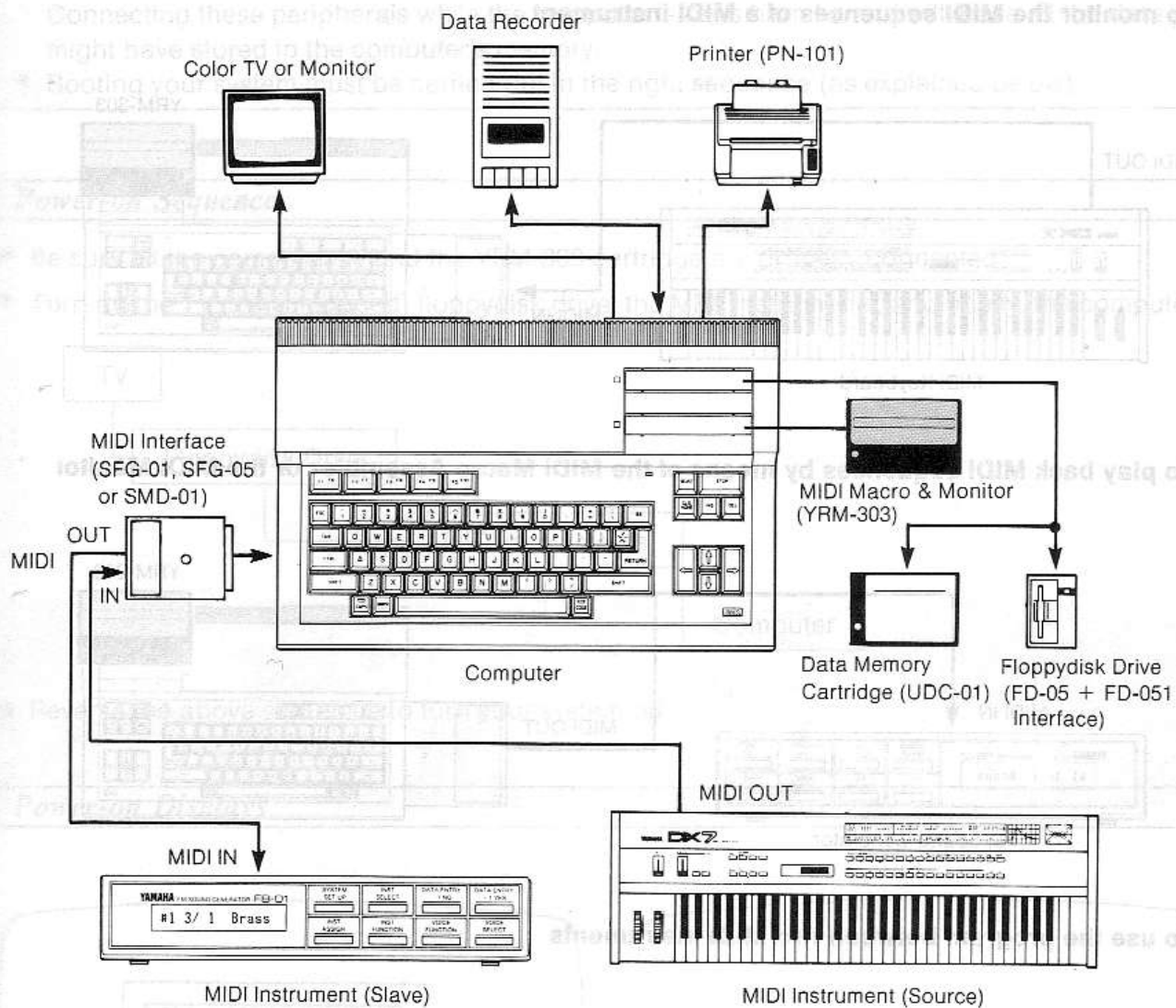
- **Source Instrument** Any MIDI unit can be used as an input device. You may use a MIDI keyboard, a MIDI sequencer or even another Music Computer equipped with sequence software, such as the FM Music Composer or the FM Music Macro, or just a Music Keyboard (YK-01/YK-20).
- **Slave Instrument** Any MIDI system featuring a MIDI IN connector. The slave instrument will play the MIDI sequences generated by or handed on from your Music Computer.

Audio System

Do not connect the AUDIO OUT jacks of your SFG-01 or SFG-05 to your sound system. Connect a stereo amplification system to the MIDI instrument(s) if they do not feature built-in amplifier and speaker.

SETTING UP YOUR SYSTEM

Overview of a Complete System



- Refer to the Owner's Manual of your computer to connect the TV/monitor and the Data Recorder.
- To connect the other components, refer to their respective Owner's Manual.
- If your computer is equipped with only one cartridge slot, you may insert the disk drive interface or the Data Memory Cartridge into the rear slot using a Single Cartridge Adapter (CA-01).

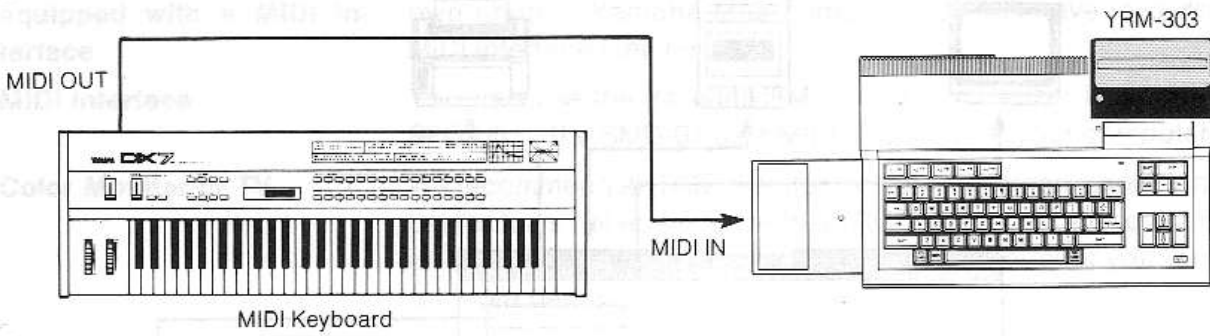
Warning:

- Never insert or remove anything from a cartridge slot as long as the computer's power indicator is lit.
- Never insert or remove a floppy disk when the drive's operation indicator is lit.
- Never switch the computer or the drive on/off while there is a floppy disk in the drive.

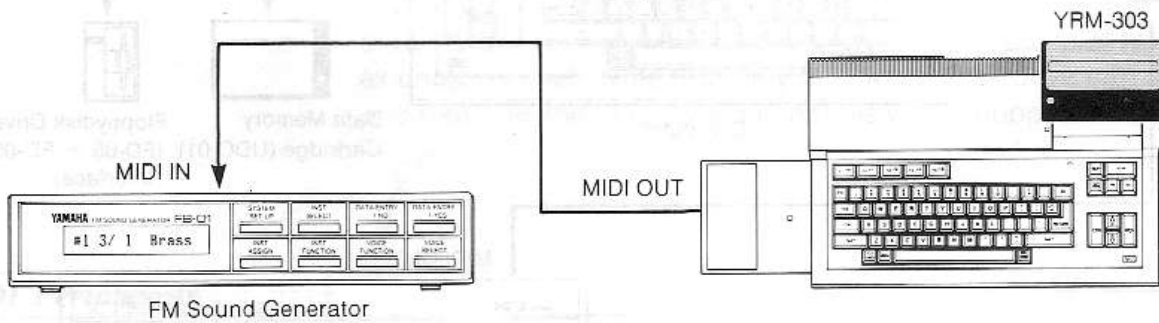
MIDI Connections

The MIDI connections depend on the way you intend to use the MIDI Macro & Monitor program.

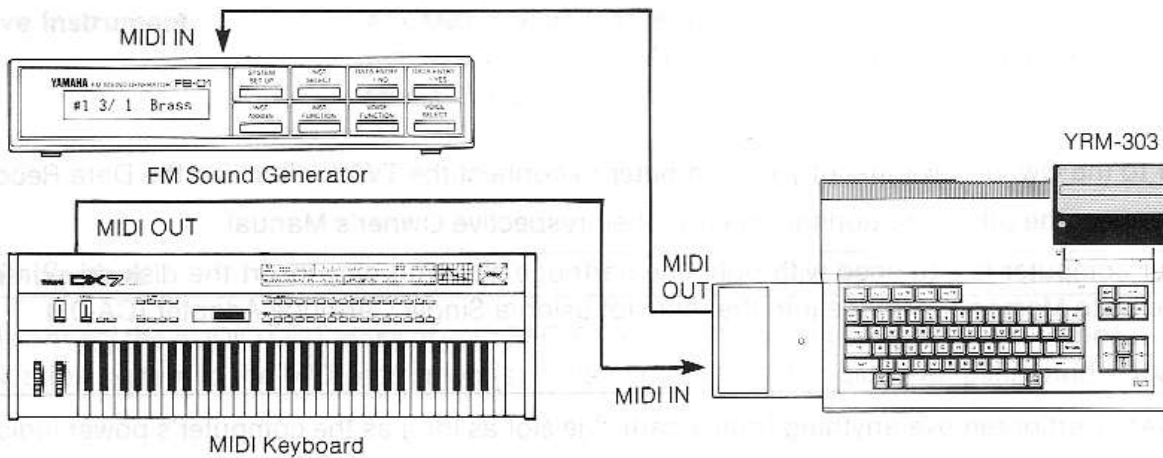
- To monitor the MIDI sequences of a MIDI instrument



- To play back MIDI sequences by means of the MIDI Macro Assembler or the MIDI Monitor



- To use the program between two MIDI instruments



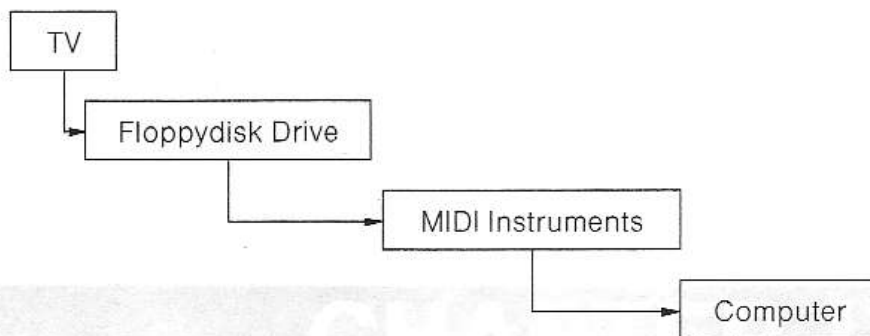
BOOTING THE SYSTEM

Important:

- Be sure to insert the YRM-303 cartridge before turning on the power to the computer. Inserting or removing a cartridge while the power is on may damage both the computer and the cartridge.
- Also insert a Data Memory Cartridge or the floppy disk drive interface before booting your system. Connecting these peripherals while the computer is on is harmful and will erase all the data you might have stored in the computer's memory.
- Booting your system must be carried out in the right sequence (as explained below).

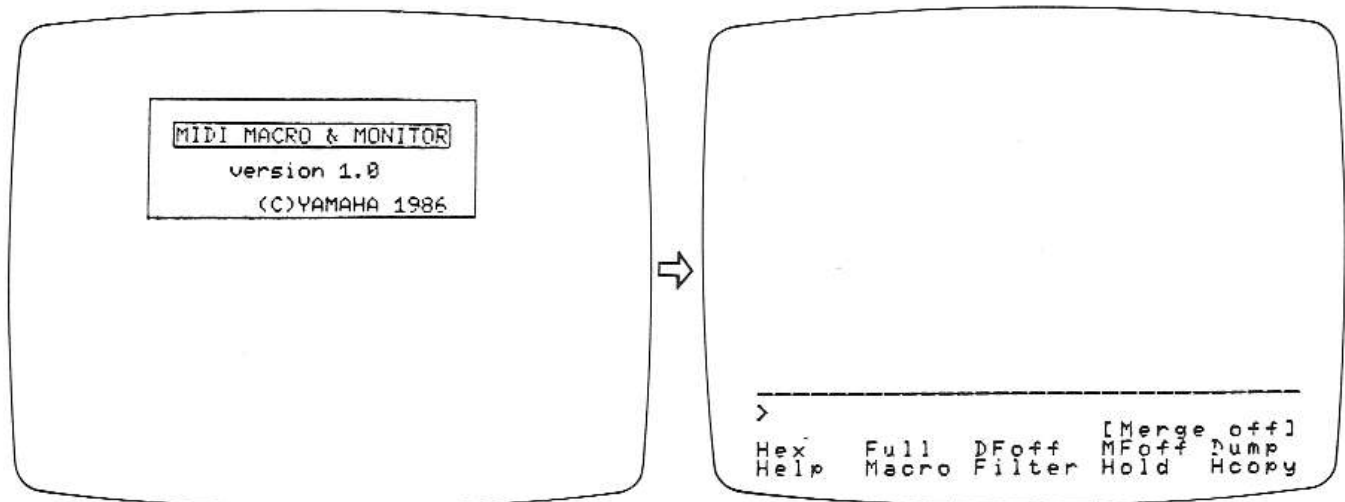
Power-on Sequence

- Be sure all the components and the YRM-303 cartridge are properly connected.
- Turn on the TV, the (unloaded) floppydisk drive, the MIDI instruments, and finally, the computer.



★ Reverse the above sequence to turn your system off.

Power-on Displays



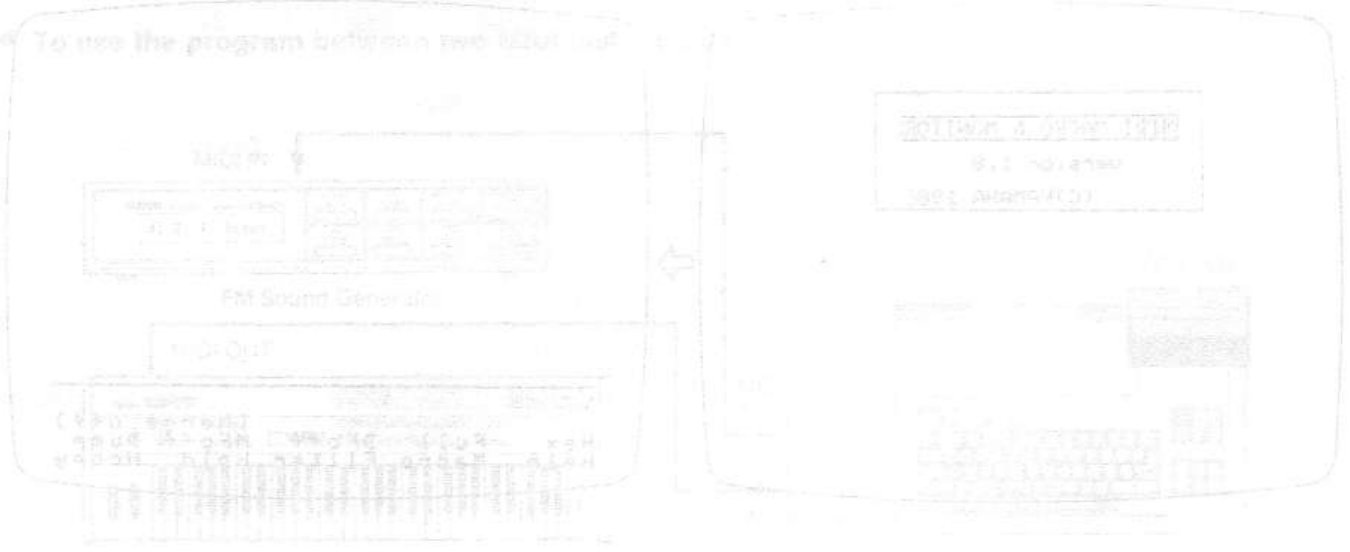
★ If the above displays fail to appear, turn off your computer and insert the cartridge correctly.

Important:

- Be sure to wait the YFM-303 cartridge before turning on the power to the computer. Inserting or removing a cartridge while the power is on may damage both the cartridge and the cartridge.
- Also insert a data floppy disk in the floppy disk drive interface before booting your system.
- Connecting these cartridges to the computer's cartridge slot is done in the same way as the data floppy disk might have stored in the cartridge's memory.
- Booting your system must be done in the right sequence (as explained below).

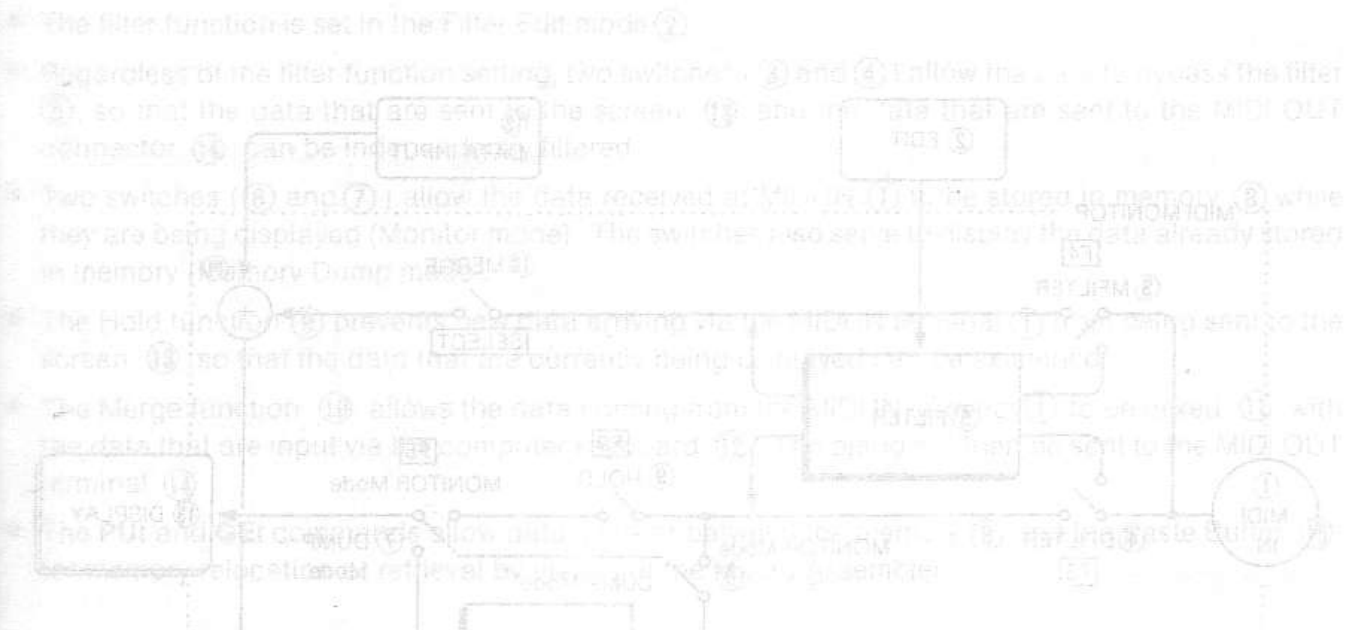


To boot your system, follow these steps:



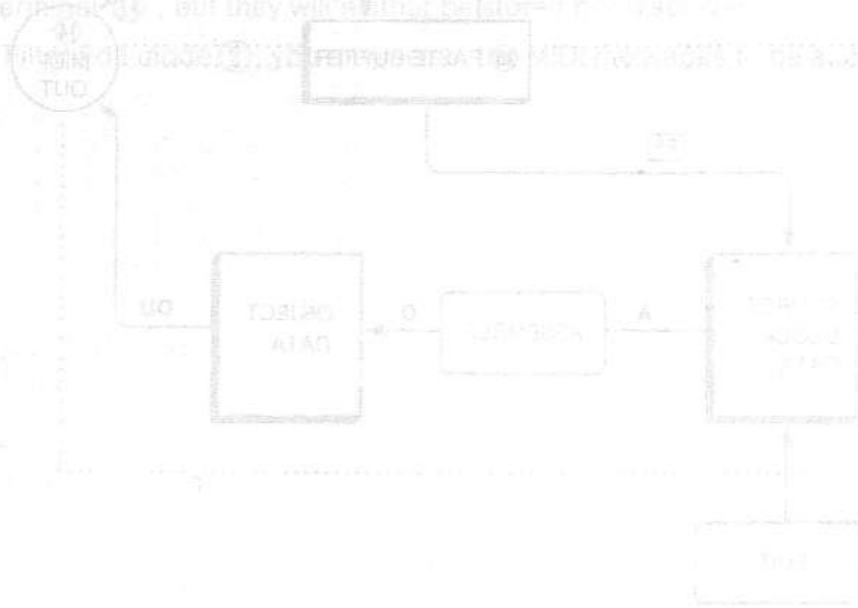
• To solve any problem, first of all, check the cartridge connection.

The following diagram shows the configuration of the MIDI Monitor. The MIDI Monitor can be configured to receive data from the MIDI IN and to send data to the MIDI OUT. The MIDI Monitor can also be configured to receive data from the MIDI IN and to send data to the MIDI OUT. The MIDI Monitor can also be configured to receive data from the MIDI IN and to send data to the MIDI OUT.



CHAPTER II THE MIDI MONITOR

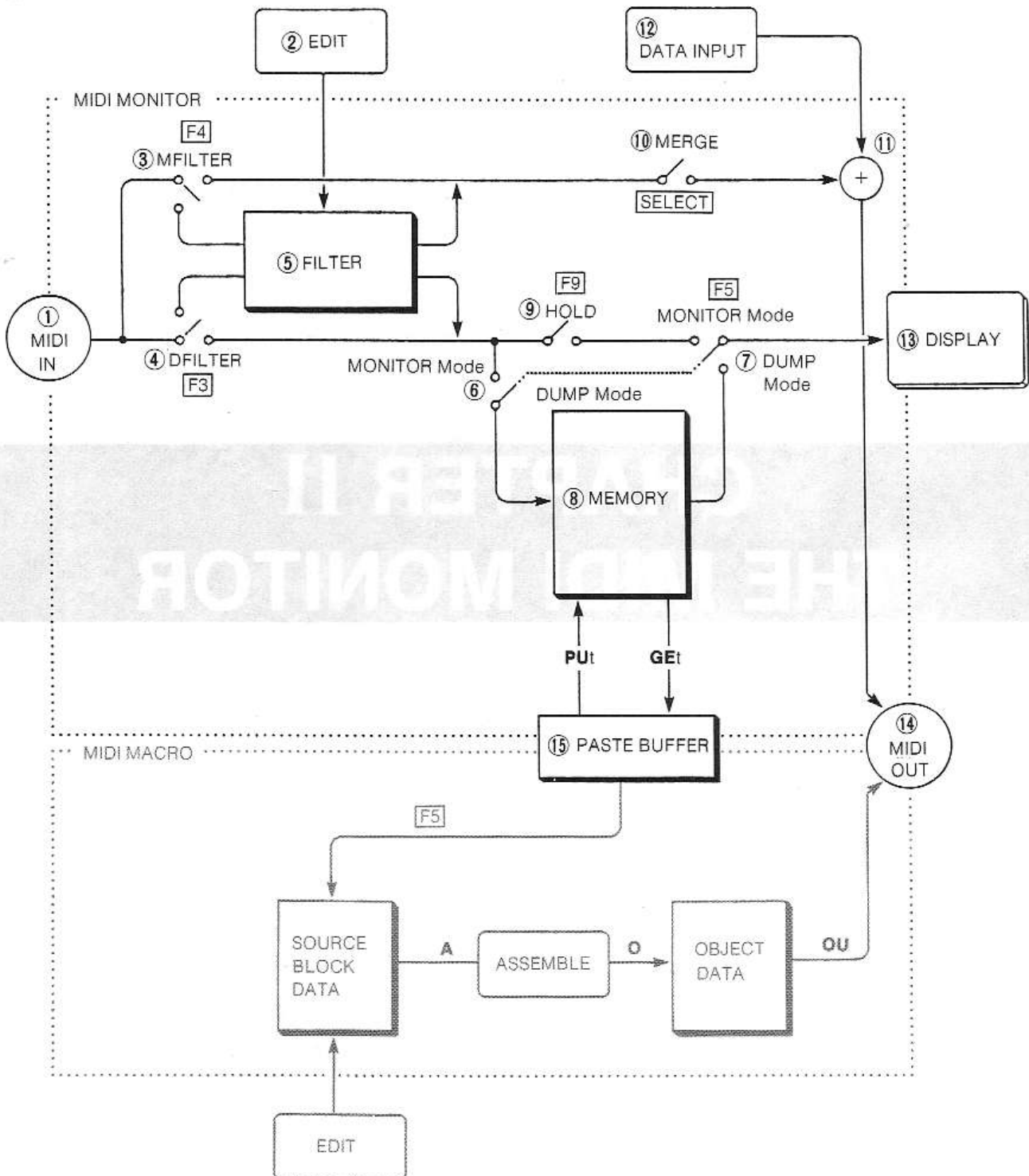
In the Memory Dump mode the data that is received from the MIDI IN is displayed and re-arranged. The data that is received from the MIDI IN is displayed and re-arranged. The data that is received from the MIDI IN is displayed and re-arranged. The data that is received from the MIDI IN is displayed and re-arranged.



A FIRST LOOK AT THE MIDI MONITOR

Block Diagram

The following diagram shows the operation of the MIDI Monitor.



Operation Outline

MIDI sequences received at the MIDI IN terminal ① are sent to the display ⑬ and to the MIDI OUT terminal ⑭ for transmission. The actual data path can be controlled by several "switches":

- The filter function is set in the Filter Edit mode ②.
- Regardless of the filter function setting, two switches (③ and ④) allow the data to bypass the filter ⑤, so that the data that are sent to the screen ⑬ and the data that are sent to the MIDI OUT connector ⑭ can be independently filtered.
- Two switches (⑥ and ⑦) allow the data received at MIDI IN ① to be stored in memory ⑧ while they are being displayed (Monitor mode). The switches also serve to display the data already stored in memory (Memory Dump mode).
- The Hold function ⑨ prevents new data arriving via the MIDI IN terminal ① from being sent to the screen ⑬ so that the data that are currently being displayed can be examined.
- The Merge function ⑩ allows the data coming from the MIDI IN terminal ① to be mixed ⑪ with the data that are input via the computer keyboard ⑫. The blend will then be sent to the MIDI OUT terminal ⑭.
- The **PUt** and **GEt** commands allow data transfer between the memory ⑧ and the Paste Buffer ⑮ for memory relocation or retrieval by means of the Macro Assembler.

The three Modes of the MIDI Monitor

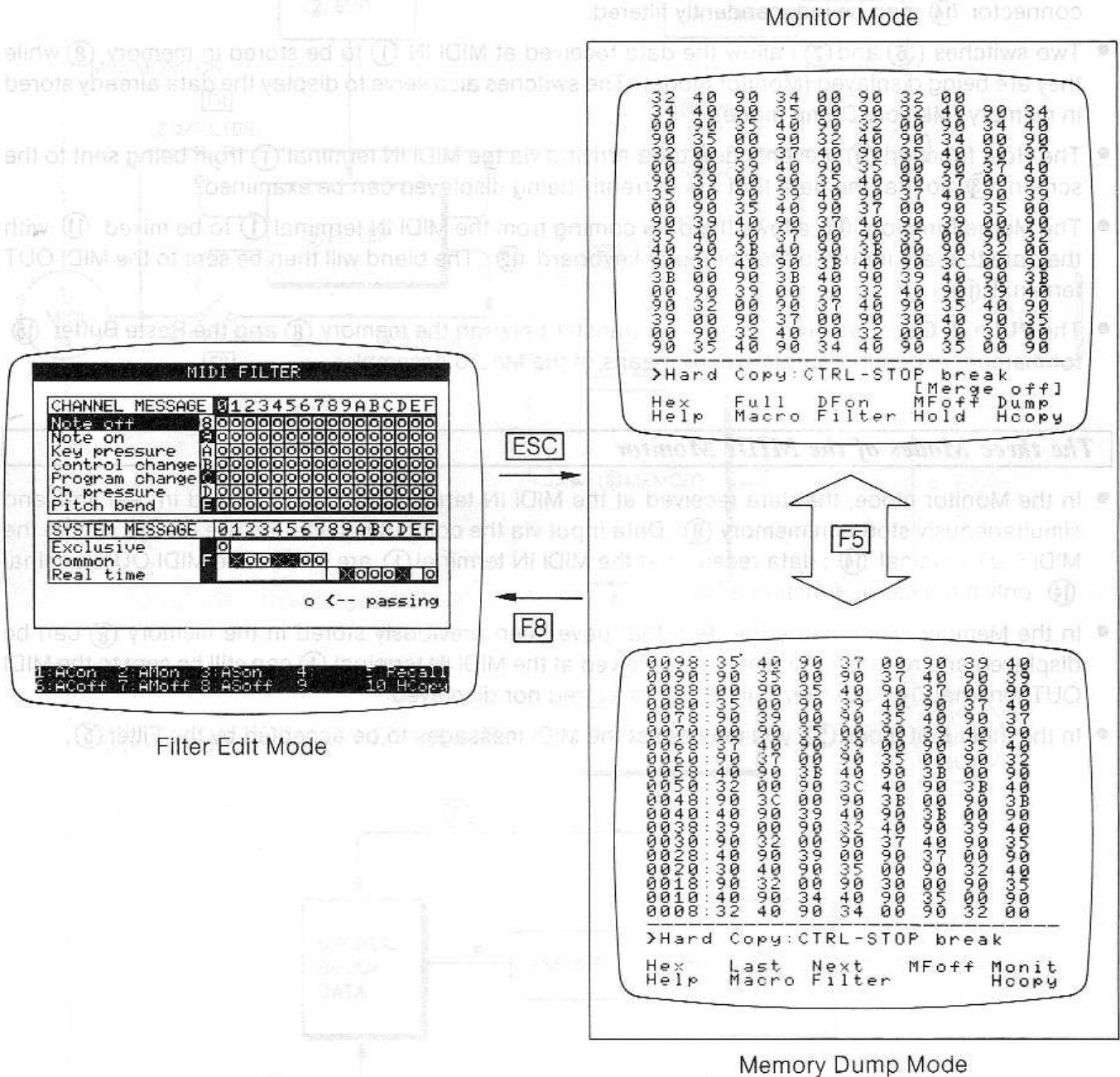
- In the Monitor mode, the data received at the MIDI IN terminal ① are displayed in real time and simultaneously stored in memory ⑧. Data input via the computer keyboard are always sent to the MIDI OUT terminal ⑭; data received at the MIDI IN terminal ① are sent to the MIDI OUT terminal ⑭ only if the Merge function is on.
- In the Memory Dump mode the data that have been previously stored in the memory ⑧ can be displayed and re-arranged. The data received at the MIDI IN terminal ① can still be sent to the MIDI OUT terminal ⑭, but they will neither be stored nor displayed.
- In the Filter Edit mode ②, you may select the MIDI messages to be accepted by the Filter ⑤.

THE MIDI MONITOR SCREENS

The MIDI Monitor features many screens: one for each of the Monitor, Memory Dump, and Filter Edit modes, plus 6 Help screens.

The following illustrations show how to access each of these screens.

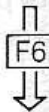
Switching the Modes of the MIDI Monitor



* The **[KANA]** key (Print Dump on/off) corresponds to the **[CODE]** key on International models.

Displaying the Help Screens

Monitor Mode



```

=====
MIDI MONITOR HELP
=====
KEY FUNCTION
=====
F1 --- Display mode Ascii/Hex
F2 --- Line Feed on/off
F3 --- Display Filter on/off
F4 --- Merge Filter on/off
F5 --- DUMP <====> MONITOR
F6 --- Help
F7 --- MACRO <====> MONITOR
F8 --- Set MIDI_FILTER
F9 --- Hold screen
F10 --- Hard copy
BS --- Back space
INS --- Insert character mode
DEL --- Delete character
TAB --- Recall command line
SEL --- Midi merger on/off
KANA --- Print dump on/off
^Z --- Click on/off
^STOP --- Stop printing
    
```

SPACE ESC To exit



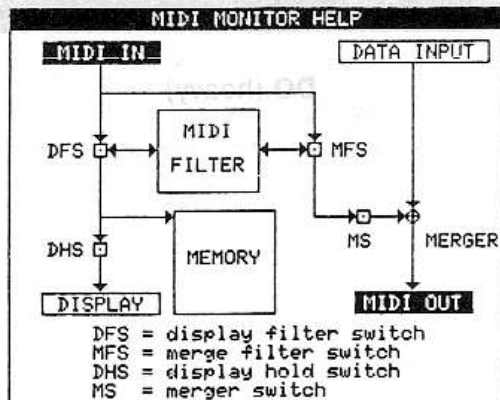
```

=====
MIDI MONITOR HELP
=====
COMMAND
=====
Clear --- clear midi input buffer
Free --- show symbol table size
List --- listing symbol table
SClear --- clear symbol table

Printer <type>,<mode>
      |
      |---> SINGLE/DOUBLE
      |---> MSX/EPSON

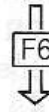
=====
TRANSMIT
=====
Symbol name is 2 characters.
The data is hex(FF) or decimal(99).
(ex) xx=12<cr>:define symbol
     xx=<cr> :delete symbol
     90 xx 40 :transmit midi data
    
```

SPACE ESC To exit



ESC To exit

Memory Dump Mode



```

=====
MIDI DUMP HELP
=====
KEY FUNCTION
=====
F1 --- Display mode Ascii/Hex
F2 --- Search Last block
F3 --- Search Next block
F4 --- Merge filter on/off
F5 --- MONITOR <====> DUMP
F6 --- Help
F7 --- MACRO <====> DUMP
F8 --- Set MIDI_FILTER
F9 --- Hard copy
BS --- Back space
INS --- Insert character mode
DEL --- Delete character
TAB --- Recall command line
SEL --- Midi merger on/off
KANA --- Print dump on/off
^Z --- Click on/off
^STOP --- Stop printing
    
```

SPACE ESC To exit



```

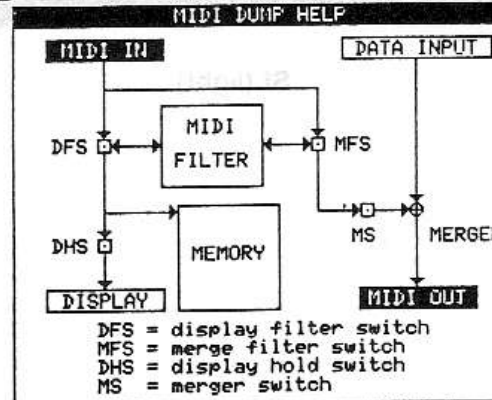
=====
MIDI DUMP HELP
=====
COMMAND
=====
Clear --- clear midi input buffer
Free --- show symbol table size
List --- listing symbol table
SClear --- clear symbol table

Printer <type>,<mode>
      |
      |---> SINGLE/DOUBLE
      |---> MSX/EPSON

GET <start_address>,<end_address>
   --- copy midi data block
   (max size 256 byte)

PUT <paste_address>
   --- paste midi data block
    
```

SPACE ESC To exit



ESC To exit

HARD COPY OF THE SCREENS

Pressing the **[F10]** (**[SHIFT]** + **[F5]**) key sends the screen data to the printer. The printout will look exactly like your screen.

- If you are not in Monitor or Memory Dump mode, press **[ESC]**.
- Before you attempt any printing, you must tell the computer which kind of printer is on line. This is done by using the **PR** command. This command accepts two parameters: the first specifies the printer; the second specifies the printing density.

Printer	MSx EPson	for MSX Printer for EPSON Printer
Density	Single DOuble	for light printing for heavy printing

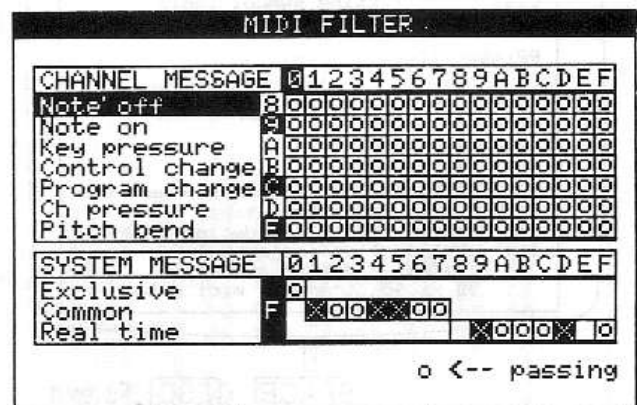
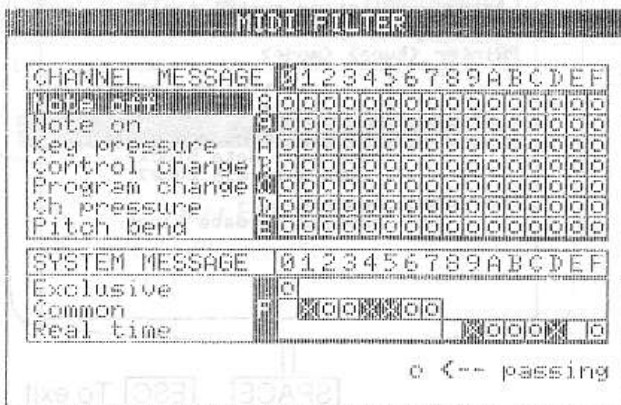
You need only to type the first two characters of these parameters.

Example:

You have an MSX printer and you want heavy print.
Type in **PR _ MS DO** and press **[RETURN]**.

- ★ In this Manual, we use the **_** symbol wherever you have to keep a blank space.
- Now, select the screen to be printed and press the **[F10]** key. The border color of the screen turns to green during printing operation and goes back to normal after completion.
- ★ If you try to print out the Monitor or Memory Dump mode screens, the border color does not turn green, but the following message appears:

Hard Copy: CTRL — STOP break



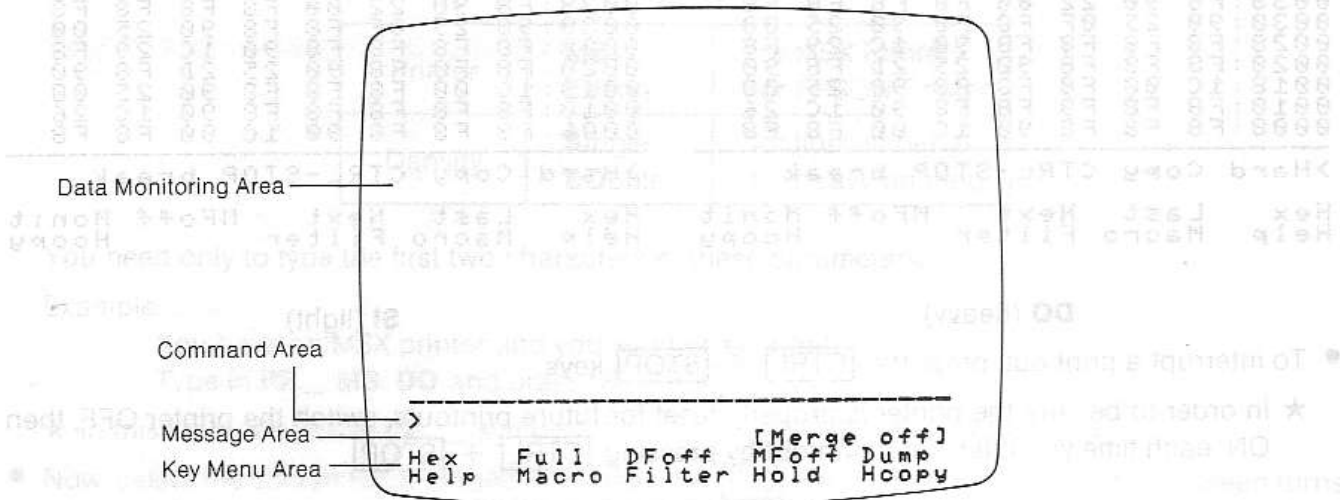
SI (light)

DO (heavy)

THE MONITOR MODE

The data received at the MIDI IN terminal can be displayed in real time, in either hexadecimal or ASCII notation. The status bytes appear on a blue background. The system is automatically set to this mode when the power is turned on.

- ★ To switch to this mode, press the **[ESC]** key if a Help screen or the Filter Edit screen is displayed. This will switch to either the Monitor or Memory Dump mode. From the Memory Dump Mode, simply press **[F5]**.



A KEY ON/KEY OFF sequence will be displayed like this:

```
90 3C 40 90 3C 00
```

The status bytes (blue background) indicate the message type. In the above example, the first data byte indicates the pitch while the second indicates the velocity (volume).

Note:

In this text we assume that the reader has some basic knowledge regarding the MIDI encoding standard. A summary of the MIDI encoding standard will be given in the Appendix of this manual. For more information, please consult the booklet "What's MIDI" published by Yamaha.

Hexadecimal & ASCII Notations

Press the **[F1]** key to toggle between hexadecimal and ASCII notation. You will usually prefer the hexadecimal notation because the display cannot be read in ASCII notation. However, the voices of the DX7 are displayed in full; BRASS for example, uses 5 characters. In hexadecimal this will read **42 52 41 53 53**. The ASCII notation is then useful to temporarily check the voice names.

Note:

In the Monitor mode, data already displayed on the screen are not properly updated by pressing the **[F1]** key. Be sure to select the Hexadecimal or ASCII notation BEFORE starting data reception.

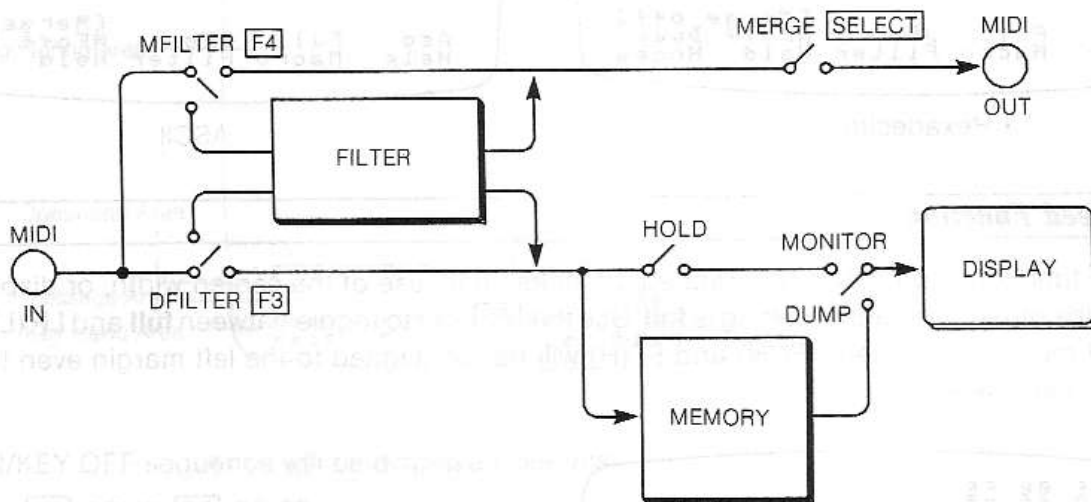
Filter Function

The setting of the filter will be explained in the "Filter Edit Mode" section of this chapter.

The filter function can be set to ON/OFF independently for the data to be displayed and for the data to be sent to the MIDI OUT connector. The **[F3]** switches the display filter ON/OFF. Depending on the setting **DFon** or **DFoff** will appear in the Key Menu area.

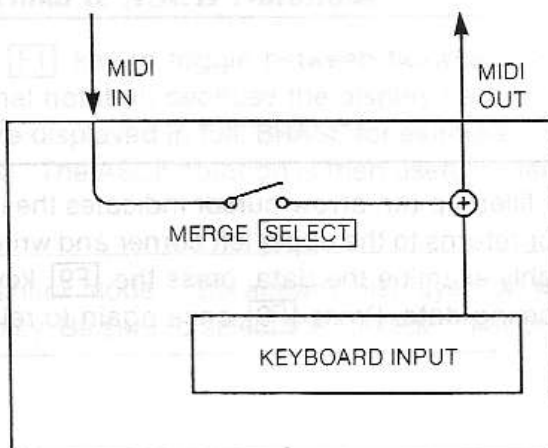
The **[F4]** key switches the MIDI OUT filter ON/OFF (only effective if the Merge function is ON). **MFon/MFoff** will appear in the Key Menu area.

If **DFon** or **MFon** are displayed, data for the display or the MIDI OUT terminal have to pass through the filter. The filter output depends on the setting you have selected in the Filter Edit mode.



MIDI Merge Function

The MIDI Merge function is switched ON/OFF by means of the **[SEL]** key. The status of the Merge function is displayed at the right side of the screen, just above the Key Menu area. The default setting is **Merge off**. If the Merge function is ON, **[F4]** is activated, and, depending on whether you selected **MFon** or **MFoff**, the data will pass through or by-pass the filter before reaching the MIDI OUT terminal. If the Merge function is set to ON, the data received at the MIDI IN connector can be mixed with the data input from the computer keyboard.



Merge off: only the data input from the keyboard are sent to the MIDI OUT connector.

Merge on: both the data input from the keyboard and the data received at the MIDI IN connector are sent to the MIDI OUT connector.

Note:

Switching the Merge function ON/OFF will clear the memory (all data are reset to 0).

Entering Data via the Keyboard

Data can be entered from the computer keyboard. These data will appear in the Command area. Pressing the **[RETURN]** key will send legal data to the MIDI OUT terminal, whereas illegal data will cause an error message to appear. Legal data constitute valid MIDI messages. The bytes of a valid MIDI message must be separated by commas (,) or spaces. The bytes may be entered in either decimal or hexadecimal notation. A decimal number must be followed by a decimal point.

Example:

F0 and **240.** are equivalent

For an example of data input via the computer's keyboard type in **C0,00** and hit the **[RETURN]** key. If the slave MIDI instrument is set to MIDI Channel 1, this MIDI message will select VOICE 1.

• Editing Functions

Some control keys will help you quickly correct a text. These keys have almost the same function as in the BASIC mode. The **[CTRL]** key, however, is not activated and cannot be used for control purposes.

Control Key	Function
[←], [→]	Move the cursor across the command line without erasing
[BS]	Drags to the left the characters located at the cursor position and at the right of this position while erasing the character at the left of the cursor.
[SPACE]	Creates a blank
[HOME]	Erases the command line.
[DEL]	Deletes the character at the cursor position and drags the following characters to the left.
[INS]	To insert characters. The color of the cursor changes to red. You can exit the insert mode by pressing either [INS] or [HOME] .

• Template Function

A template function is provided to further ease the input of data. Each time you enter a line of data, the line is stored. Pressing the **[TAB]** key will cause the same line to be displayed again.

• Creating Macro Symbols

It is possible to assign data to symbolic variables, so that these symbols stand for the assigned value in further input. Such a definition uses the equal (=) sign.

Example:

Type in **V = B0, 07** and press **[RETURN]**.

The following message appears:

V is defined: length = 02

The length is the number of bytes affected to the symbol **V**. If you forgot the comma (or space) between the two bytes, the length will be 1 and **07** will be dropped.

★ A re-definition of the same symbol will erase the old one.

★ A previously defined symbol can be used to define other symbols.

Example:

W = V, 7F will assign the value **B0, 07, 7F** to the symbol **W**.

Important:

The legal symbols are submitted to restrictions similar to the restrictions undergone by the BASIC variables:

- A symbol is made of up to two characters, the first being a letter of the English alphabet and the second a letter of the English alphabet or a numeric.
- The name of existing commands like **LI**, **AA** are illegal symbols (refer to the list of commands to avoid using conflictive symbols).
- Upper-case and lower-case letters cannot be distinguished (**M** and **m** are not two different symbols).
- As the characters from **A** to **F** are used as hexadecimal digits, they cannot be used in combination with a numeric or another character from **A** to **F** to define a symbol (**AA**, **F9** are invalid; **AV** is valid).

• Listing and Erasing Symbols

A re-definition of a symbol takes precedence over the old definition. To erase a symbol, take the following steps:

Type in the **LI** command and press **[RETURN]**. The list of defined symbols will be displayed (to exit the list display, press any key).

To erase a registered symbol, **PC** for instance, type in **PC =**, and press **[RETURN]**.

To erase all symbols, type in the **SC** (symbol clear) command, and press **[RETURN]**.

Erasing symbols is sometimes necessary to free up some memory space in order to define other symbols.

To know how much space is left in the symbol memory, type in the **FR** command and press **[RETURN]**. A message will indicate how many bytes are still available.

The space necessary to store the definition of a symbol obeys the following formula:

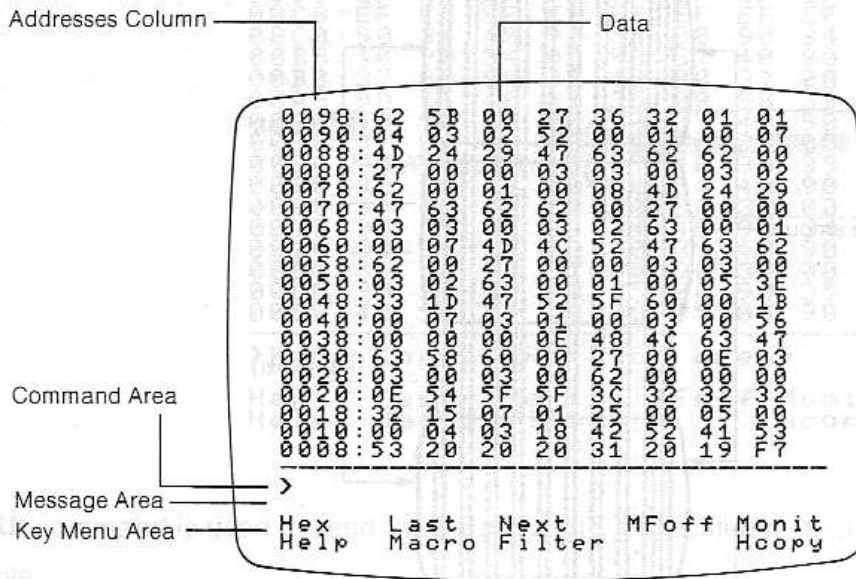
$$\text{Symbol name (2 bytes) + Length of data (1 byte) + one byte per data}$$

Note:

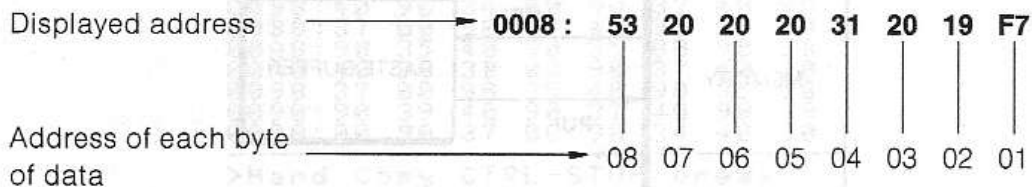
Keyboard input can sometimes be shortened when the same status byte has to be used in consecutive messages. For details, refer to the Appendix F, last note on "RUNNING STATUS".

THE MEMORY DUMP MODE

The **[F5]** key is used to toggle between the Monitor and Memory Dump modes.



The Memory Dump mode is used to display the data stored in the memory. When this mode is set, the data received at the MIDI IN terminal are prevented from going into the memory or from being displayed. The Memory Dump screen shows the data from the last input. Scrolling up and down can be carried out by pressing the **[↑]** and **[↓]** cursor keys. The data stored in the memory, before the Memory Dump mode is turned on, start at the address 0001. Each time new data are stored in the Monitor mode, the old data are pushed up so that the new data can be stored at the bottom of the memory. The total capacity of the memory is 4336 bytes. An address of the Address Column indicates the highest address of the corresponding line of data, that is, the address of the leftmost data.:



Hexadecimal & ASCII Notation

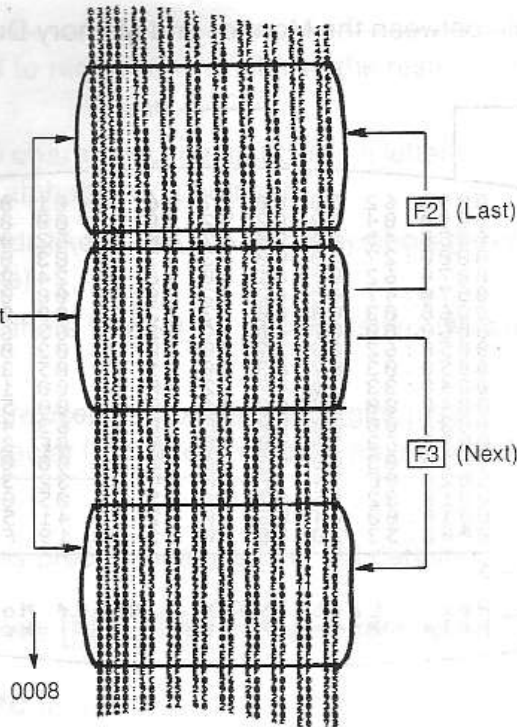
The **[F1]** key is used to toggle between hexadecimal and ASCII notation. Refer to the former paragraph (The Monitor Mode) for further details.

Building a Block of Data

When switching from the Memory Dump mode to the Monitor mode, reception of new data starts. A string of 24 EF(H) codes is, however, inserted between the new and the old data. As a reception of 24 consecutive EF(H) codes never occurs in MIDI communication, these codes will constitute a good separator and make it easy for you to distinguish successive blocks of MIDI messages.

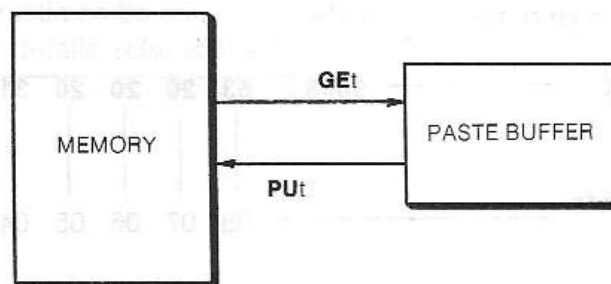
Moreover, the **[F2]** and **[F3]** keys can be used in the Memory Dump mode to scroll the screen up and down by the amount of one block.

Scrolling by the amount
of one block



Relocating the Data

The **GEt** and **PUt** commands are used to re-arrange data in the memory. Relocation of data is carried out in two operations: first you transfer data from the memory to the Paste Buffer, next you send back the data from the Paste Buffer to the memory.



- The **GEt** command is used to select a block of data from the memory and store it into the Paste Buffer. A maximum of 256 bytes can be transferred at a time.

Example:

Type in **GE _ 8D, 95** and press **[RETURN]**.

★ A space must always be left between a command and its parameter(s). Parameters may be separated by a comma or a space.

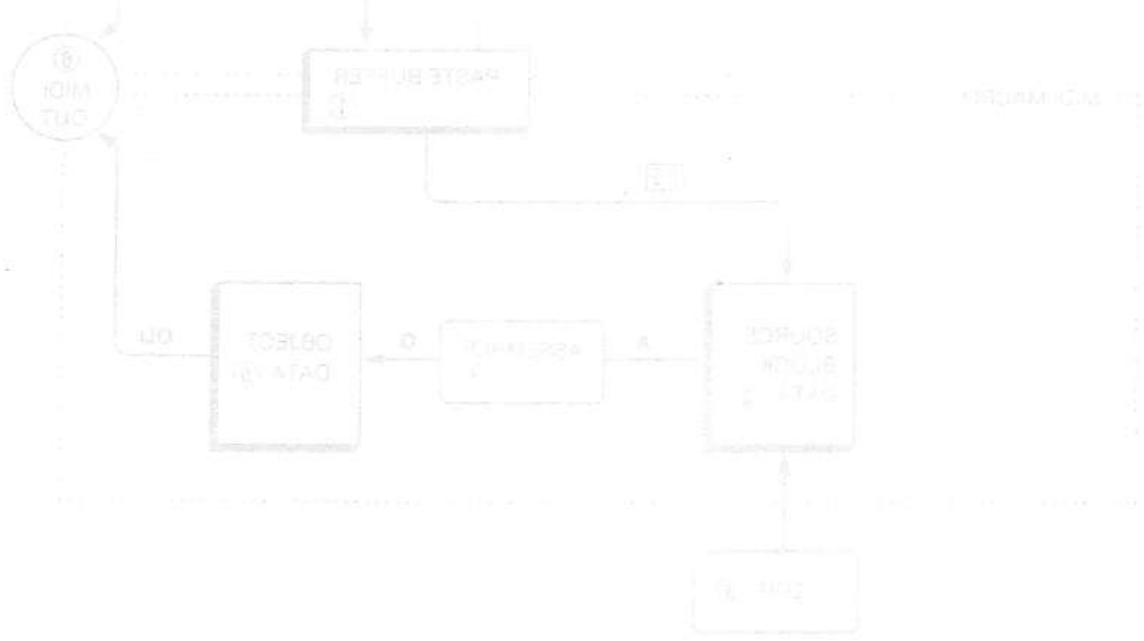
The two parameters of the **GE** command indicate the first and last address of the data to be stored into the Paste Buffer.

The following diagram shows the operation of the MIDI Macro Assembler. It starts in view state and reads MIDI commands.

The MIDI Macro Assembler has three modes: Command mode, Edit mode, and Run mode. In command mode, the user enters commands.



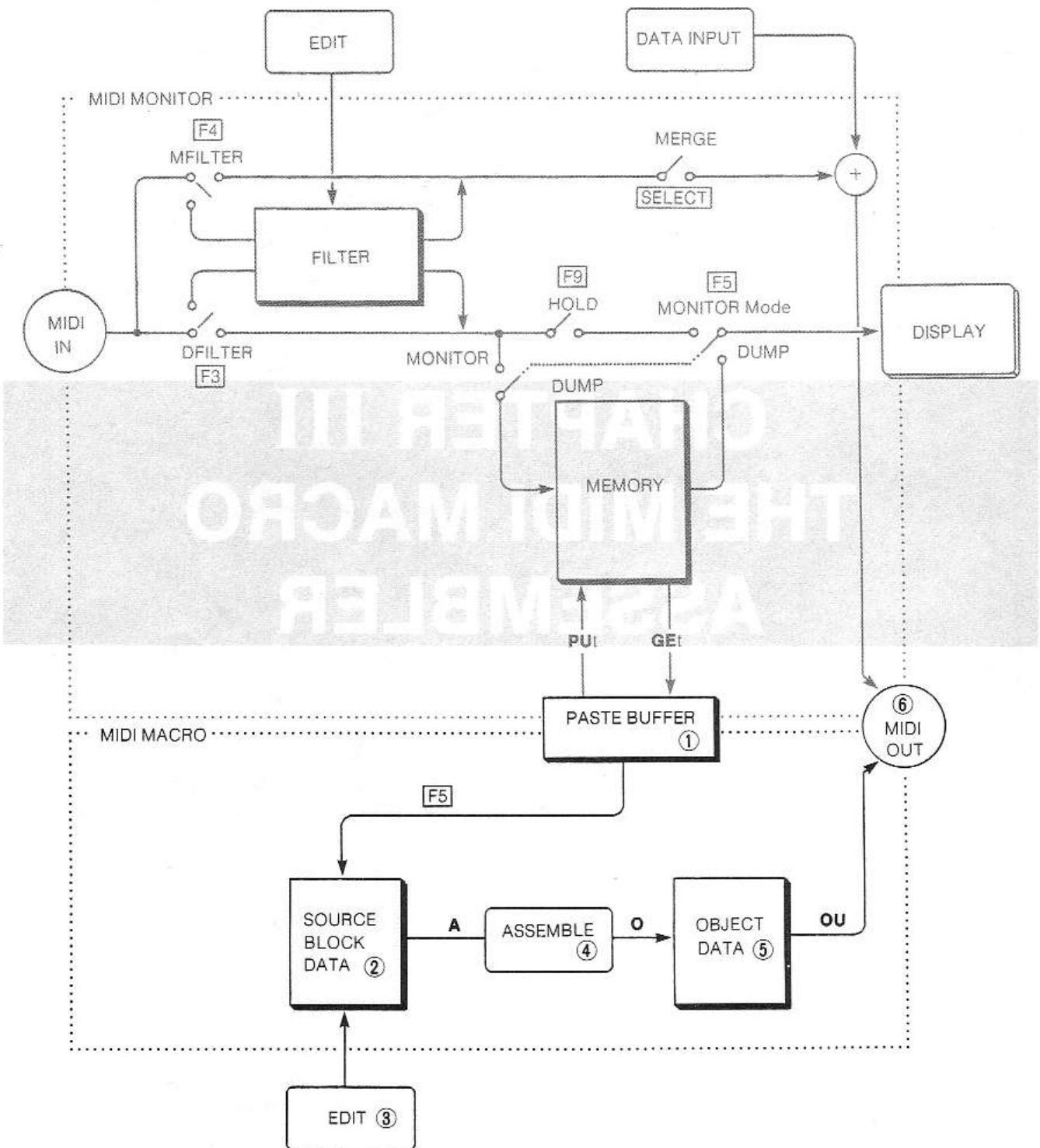
CHAPTER III THE MIDI MACRO ASSEMBLER



A FIRST LOOK AT THE MACRO ASSEMBLER

Block Diagram

The following diagram shows the operation of the MIDI Macro Assembler.



Outline of the Operation

The MIDI Macro Assembler is an efficient tool for creating MIDI data blocks.

There are three ways to start creating MIDI data:

- You may create the source data from scratch in the Edit mode ③.
- You may retrieve an old block of data previously saved on cassette tape, Data Memory Cartridge or floppy disk.
- You may first create rough data using the MIDI Monitor and transfer these data into the Paste Buffer. The MIDI Macro then allows you to transfer these data into the Edit memory for further editing.

The MIDI Macro Assembler has three modes:

- In the Edit mode, you may either edit your data or define the macro symbols you want to use for easy writing and understanding of the source data ②.
- In the command mode, you have access to functions such as Assemble ④ (performing the conversion source data → object data) or Output (sending the object data ⑤ to the MIDI OUT terminal ⑥).
- The File mode is used for saving/loading either the source or object data. The object data saved in this mode can be used with the MIDI Recorder (YRM-301) or the RX Editor (YRM-302).

Important:

In this manual, "source data" refers to the data you actually edit, while "object data" refers to the data that can be actually transmitted to a MIDI instrument. The source data consist of MIDI messages, symbols standing for MIDI messages (you may define these symbols in the Edit mode), and comments you might wish to insert for easier understanding. The object data consist of pure MIDI messages in hexadecimal notation.

THE MIDI MACRO ASSEMBLER SCREENS

The following illustration shows how to switch to the different screens of the MIDI Macro Assembler.

Edit Mode

MIDI MACRO ASSEMBLER			SRC OBJ
[BLOCK INFORMATION]			Font
No.	Name	Size	
01	xxxxxxx	77	
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			

E or S ↑ (ESC) ↓ F1

Command Menu
(Help Screen)

MIDI MACRO ASSEMBLER		SRC OBJ
COMMAND MENU		
Sedit	:	edit global symbol
Edit	(m):	edit block
Object	(m):	list object data
Assemble	(m):	midi assemble
ASsemble	:	all midi assemble
OUT	(m), (s):	midi out
Lout	(m), (s):	looping midi out
SOut	(m):	single step midi out
(O)Directory	:	directory block
(O)Kill	m	:kill block
(O)Copy	m,n	:copy [m] to [n]
(O)Swap	m,n	:swap [m] and [n]
(O)Print	m,n	:print from [m] to [n]
m,n= block# / s= trace step (0..40)		

F8 ↓ ↑ ESC

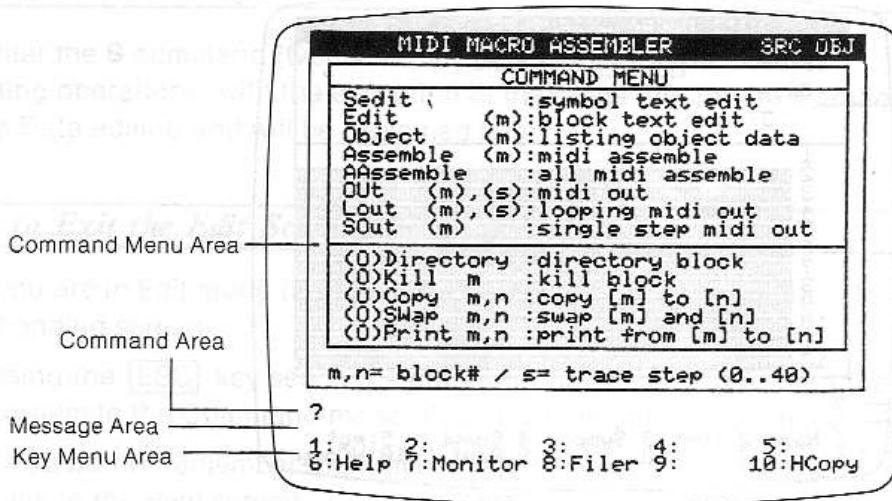
File Mode

FILER	
Type	File: [.MMS]
Source	
Object	
Device	
Disk-A	
Cassette	
DC	
Operation	
Files	
NextFile	
Load	
Save	
Kill	

These screens can be printed out by using the **[F10]** key. Refer to Chapter 2, "Hard Copy of the Screens".

THE COMMAND MODE

When you switch your system from the MIDI Monitor to the MIDI Macro Assembler (by pressing the **F7** key), you are in the Command mode of the MIDI Macro Assembler. This mode displays a list of the available commands.



Available Operations

In the Command mode, the following operations can be carried out:

- Source data assembly
- Object data output
- Object data display
- Data block management:

- Screen display
- Delete
- Copy
- Swap
- Printout

How to Enter a Command

The left column of the Help Menu indicates the available commands and which parameter(s) you may enter along with the command. The full name of each command is displayed only for easy understanding: when entering a command, just type the upper-case letter(s) of the command.

Example:

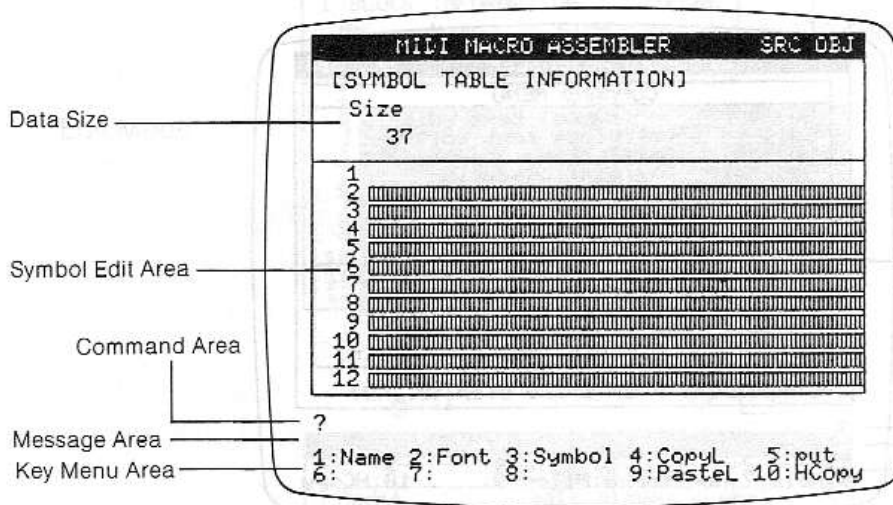
Sedit (Symbol Edit): type in **S**.

A space between a command symbol and its first parameter is mandatory. When a command requires two parameters, insert a comma between the parameters.

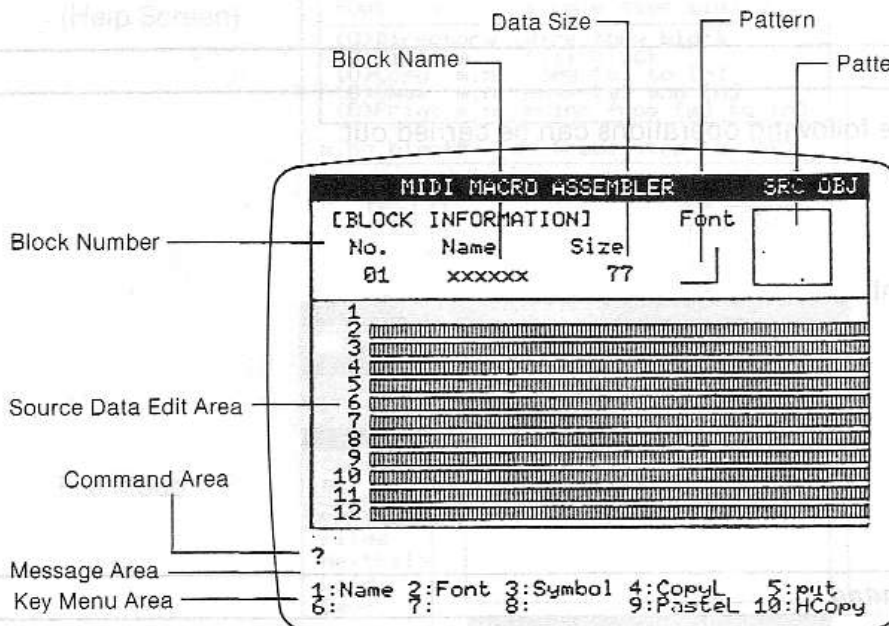
After typing a command and its parameter(s), press the **RETURN** key to enter the command.

THE EDIT MODE

To switch from the Command mode to the Edit mode, enter the **S** or **E** command. **S** is used when you want to edit the macro symbols to be used in your source data; **E** is used for actual editing of the source data.



Symbol Edit Screen



Source Data Edit Screen

Editing the Source Data

To edit a block of source data, enter the **E** command followed by a blank space and the number of the block you want to edit.

Example:

E_6 [RETURN]

The block numbers range from 1 to 16, and the selected block number will be displayed on the Source Data Edit screen.

If you omit the block number, this specification will default to the last edited block (or block #1 if you enter the **E** command for the first time).

Editing the Macro Symbols

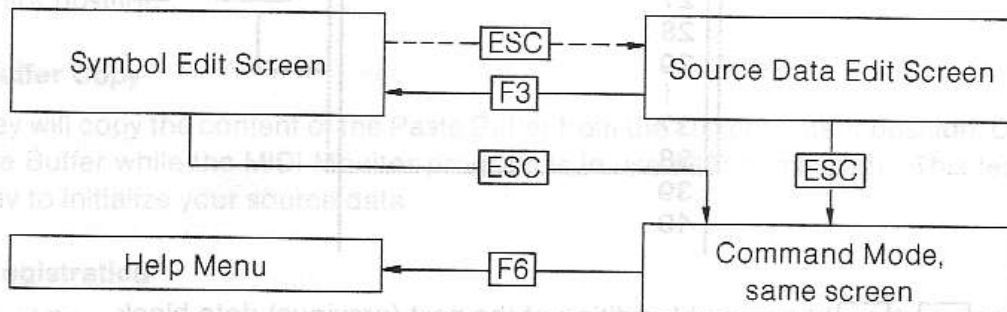
Just enter the **S** command (type in **S** and press **RETURN**).

All editing operations (with the exception of the name and font registration) are carried out as for the Source Data editing and will be explained later.

How to Exit the Edit Screens

When you are in Edit mode (Source data or Symbol Edit), the cursor is located in the Edit Area of the corresponding screen.

- Pressing the **ESC** key sends the cursor to the Command Area of the same screen and switches the system to the Command mode. You may then enter a command.
 - ★ If you do not remember the name of the command you want to enter, press the **F6** key to go back to the Help screen.
 - ★ If you mistakenly pressed the **ESC** key while editing Macro symbols for instance, just enter the **S** command again.
- Pressing the **ESC** key while editing Macro symbols will switch the system to the Source Data Edit screen (see Note below).
- Pressing the **F3** key while editing Source data will switch the system to the Symbol Edit screen.



Note:

The effect of the **ESC** key during Symbol Editing depends on how the screen was accessed:

- Accessed by the **S** command → Back to Command mode.
- Accessed by the **F3** key (from the Source Data Edit screen) → Back to the Data Edit screen.

Keyboard Operation in Edit Mode

The following explains the use of special keys that make it easy to edit data in either Source Data or Symbol Edit mode.

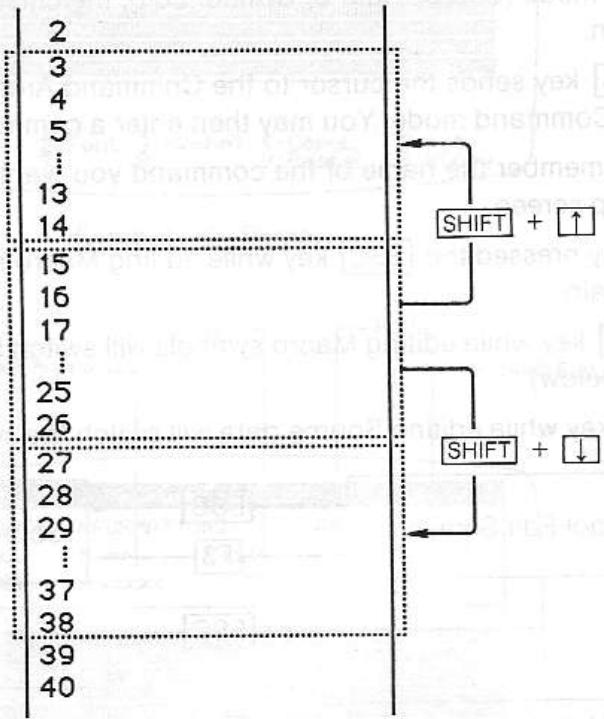
• **Cursor Movement**

Use the cursor keys **↑**, **↓**, **→**, and **←**. The cursor will move in the directions indicated by the arrow-mark, except in the following circumstances:

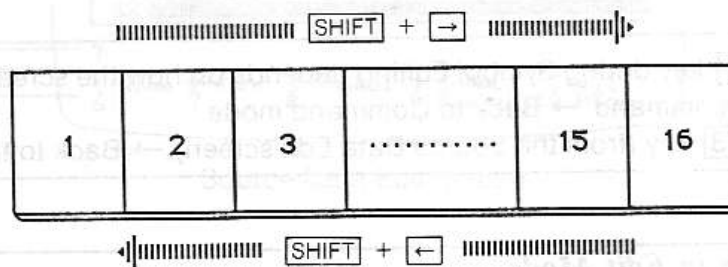
Key	Cursor will not respond if ...
→	at the end of the last sentence of a block
←	at the beginning of the first sentence of a block
↑	in the first sentence of a block
↓	in the last sentence of a block

• **Scroll-up/Scroll-down**

The [SHIFT] + [↓] ([↑]) keys will move the cursor to the beginning of the next (previous) page.



The [SHIFT] + [→] ([←]) keys enable editing of the next (previous) data block.



• **Back Space**

Pressing [BS] will delete one character to the left of the cursor and drag the characters of the line from the cursor position to the end of the line back one character space to the left.

• **Delete**

Pressing the [DEL] key will delete the character the cursor is over and drag the rest of the line one character to the left.

Pressing the [SHIFT] + [DEL] keys will delete all the characters of the line and move the following lines one position up.

Pressing **CTRL** + **E** will delete all the characters of the line to the right of the cursor position.

- **Insert**

Press the **INS** key to toggle between the insert and write-over modes. In insert mode, the cursor is yellow. Typing one character in insert mode displays the character at the cursor position and moves the rest of the line — and the cursor — one character to the right.

Pressing **SHIFT** + **INS** creates a blank line and moves the lines one position down from the cursor position.

- **Line Break**

Pressing the **RETURN** key in the middle of a line moves the rest of the line to the beginning of the next line, and the subsequent lines one position down.

Use of the Function Keys

- **Switching from the Source Data Edit Screen to the Symbol Edit Screen.**

Pressing **F3** switches from the Source Data Edit screen to the Symbol Edit screen. When the Symbol Edit screen is accessed that way, pressing the **ESC** key will not activate the Command mode but switch back to the Source Data Edit screen.

- **Line Copy/Paste**

Pressing the **F4** key stores the current line. Pressing the **F9** key will write the stored line from the current cursor position.

- **Paste Buffer Copy**

The **F5** key will copy the content of the Paste Buffer from the current cursor position. Data are stored in the Paste Buffer while the MIDI Monitor program is in use (**GEt** command). This feature provides an easy way to initialize your source data.

- **Name Registration**

Each block of Source data can be assigned a name. This name will be used by the Yamaha MIDI Recorder or RX Editor to select the file to be loaded.

Pressing the **F1** key in the Source Data Edit mode moves the cursor to the Block Name Area. Type in the name (6 characters) and press the **RETURN** key.

★ Pressing **ESC** instead of **RETURN** cancels the name input and restores the old name.

★ The following characters are available for the block name:

A ~ Z, a ~ z, 0 ~ 9

- **Pattern (Font) Registration**

Each block can be assigned a special pattern. When you insert a MIDI Macro in the Chain Mode of the Yamaha MIDI Recorder (YRM-301), this special pattern will be displayed so that you can easily locate and identify the MIDI Macro.

Pressing the **F2** key enables the Pattern Editing. Use the cursor keys to move the cursor across the big box at the upper-right corner of the screen. Use the space bar to switch one dot ON/OFF. While you edit the pattern, a real-size character is displayed at the left of the big box. Press the **RETURN** key to enter the pattern.

- ★ The size of the Pattern is 16x16 bits. However, the right vertical line and the bottom horizontal line are used as separators for the MIDI Recorder's display. The box perimeter may be edited but the abovementioned lines will be reset to their default when displayed by the MIDI Recorder.

The following keys feature interesting editing functions:

Key	Function
HOME	90° rotation, counterclockwise.
INS	color reverse
DEL	vertical reflection of the pattern
SHIFT + ↓, ↑, ←, →	translation in the direction of the arrow (32 x 32 bit period).
TAB	initialization (only the two abovementioned lines appear).

OPERATION GUIDELINES

Let's try out a few simple examples to illustrate how the MIDI Macro Assembler is working.

System Settings

In the following examples we assume that a DX7 — or other MIDI keyboard — and a Yamaha FB-01 are used as source and slave instruments respectively.

Any system can be set by using System Exclusive Messages. However, in order to make it easier for you to use the following examples, even if your system is not the abovementioned system, we will avoid using System Exclusive messages here and carry out the initial settings by a sequence of MIDI messages.

★ Before starting with these examples, set the slave instrument (FB-01) to the reception MIDI Channel 1.

We want to carry the following settings:

Voice #	6
Volume	110
Modulation Wheel	0

Preparing the MIDI Messages

We are going to create a simple macro, made up of the following messages:

- Program Change (Voice Change)
- Volume Change
- Modulation Wheel Control

So let's first analyze what these messages must consist of.

• Program Change

The Program Change message is made of two bytes: a status byte identifying the rest of the message and specifying the MIDI Channel, and a data byte specifying the new voice.

Status: 1100nnnn or Cn(H)
Voice: 0ppppppp

Since the FB-01 is set to reception channel 1, $n = 0$, and the status bytes must be C0(H) or 192(D).

★ MIDI Channels are numbered from 1 to 16, which correspond to the actual four-bit specifications from 0 to 15.

The data byte must be 5 if we want to switch to voice #6. The complete program Change message will be (in hexadecimal notation):

Program Change: **C0 5**

• Volume Change

The Volume Change is one of the Control Change Messages and is made up of three bytes: a status byte identifying a Control Change Message and specifying the MIDI channel, a data byte identifying the specific control (here volume change), and finally a data byte specifying the new value.

Status: 1011nnnn or Bn(H)
Control #: 0cccccc
Value: 0vvvvvvv

Since the MIDI channel is 1, the status byte is B0(H). A volume change corresponds to Control Change #7 so that the second byte is 7.

To set the volume to the value of 110, the third byte must be 6E(H) or 110(D).

The complete volume change message will be (in hexadecimal notation):

Volume Change: **B0 7 6E**

• Modulation Wheel Control

The Modulation Wheel Control is another Control Change. Its number is 1. The value byte must be 0 if we want to set the Modulation Wheel to 0. The complete modulation Wheel Control will be (in hexadecimal notation):

Modulation Wheel Control: **B0 1 0**

The sequence of MIDI messages necessary to obtain the desired settings is then:

C0 5
B0 7 6E
B0 1 0

Source Data Construction

• Starting the Editor

If the MIDI Monitor program is enabled, press the **[F7]** key to switch to the MIDI Macro Assembler program.

If you are not in the Command mode of the MIDI Macro Assembler, press the **[ESC]** key once or twice, until you get the Command mode.

We are going to write the above data in block 1. Enter the **E _1** command.

Type in **E _1** and press **[RETURN]**.

If you already have some data in block 1, select another block.

• Data Input

Each data should be written in sequence, from C0 to 0. Hexadecimal data must be preceded by the **\$** symbol; decimal data can be written in the usual way. A data must be separated from the following data by a comma.

After the input of a complete message, press **[RETURN]** to go to the next line. In the case of a mistype, use the edit keys (**[BS]** for example).

After the input of the three messages your screen will appear as follows:

MIDI MACRO ASSEMBLER				SRC	OBJ
[BLOCK INFORMATION]				Font	
No.	Name	Size			
01	xxxxxxx	149			
1	\$c0,5				
2	\$b0,7,110				
3	\$b0,1,0				
4					
5					
6					
7					
8					
9					
10					
11					
12					

This concludes the data input procedures. The text displayed on the screen is what we call a source data.

• **Assembling Data**

The source data cannot be directly sent to the MIDI OUT connector: you must first convert the source data into object data. This conversion is referred to as assembling.

To assemble your data, press **[ESC]** to enable the Command mode. You may then press **[F1]** to display the list of the available commands. Assembling is carried out by the **A** command.

Type in **A _ 1** and press **[RETURN]**.

To display the object data, use the **O** command.

Type in **O _ 1** and press **[RETURN]**.

Note that all the object data are now displayed in hexadecimal notation.

• **Transmitting the Object Data**

Press **[ESC]** to enable the Command mode and use the **[F6]** key to go back to the Help Menu.

Transmission of the object data is carried out by the **OU** command. This command accepts two parameters. The first parameter is the block number specifier; the second one will be explained later.

Type in **OU _ 1** and press **[RETURN]**.

MORE ABOUT DATA INPUT

MIDI data consist of sequences of MIDI messages — a MIDI message being made of a status byte and one or more value bytes. Status and value bytes are nothing but numbers from 0 to 127 (value bytes) or from 128 to 255 (status bytes). Source data will therefore appear as a sequence of numbers. The MIDI Macro Assembler, however, accepts many different ways for writing these numbers, so that you may choose the most convenient one for each data to be input. You may input numbers, as we did in the above example, or write symbols standing for numbers. You may even use operations to generate numbers. In many instances all these features result in source data and object data that look very different: the object data are pure numbers, displayed in hexadecimal notation while the source data will appear as a symbolic text, very much easier to understand. For the source data input you will use what we call here a macro language — a system of symbolic writing easier to handle than pure numbers. When you activate the **A** command, your symbolic source text will be correctly converted into pure numbers (the object data ready for transmission), provided the program is able to understand the meaning of your source text. The rules governing a correct input of the source data are what we call here, the syntax of the macro language. If you respect the syntax rules your source text will be properly assembled, if not, assembling will be impossible, or will produce unwanted results. The syntax is the subject of this section. We recommend you carefully read the following explanations.

Input of Constants

A constant is a numeric value. In the above example, our source text was created by input of constants only. As you noticed, we used both decimal and hexadecimal notation, according to which notation was the most convenient. There are actually four legal notations for the constants.

• **Decimal Notation**

Simply write the number as you usually do: using digits from 0 to 9 without any prefix. Decimal notation is the most convenient for values like volume setting or control numbers.

Example:

\$B0, 7, 110 (7 = volume control #; 110 = setting value)

• **Hexadecimal Notation**

The table of MIDI messages usually gives the status byte in hexadecimal notation. This notation will therefore be most useful when writing a status byte.

A hexadecimal constant starts with the **\$** prefix and uses two digits, from 0 to 9 or A to F.

Example:

\$B0, 7, 110 (\$B0 = Control Change status byte for MIDI Channel 1)

★ A Channel message status byte is most easily interpreted in hexadecimal notation: the first "digit", "B" in the above example, announces a Control Change; the second digit, "0", indicates that channel 1 is affected by this message.

• **Binary Notation**

Binary notation uses the **%** prefix, and is made of 8 digits (1 or 0). This notation should be used only for very specific data, where each single bit is assigned a special meaning. To turn the operators of the FB-01 ON/OFF for example, you need a value that has the following binary structure:

% 0**000** (* = relevant bit; 0 = unused bits)

The order of the operators is 4, 3, 2, 1

Example:

% 01100000 (turns ON OP4, OP3 and turns off OP2, OP1)

In decimal notation this value will read 96 — and is hard to relate with the desired effect.

• Key Name

This will be used for pitch indication. If you want to set a pitch to A#1, for example, in a Note ON message, you may of course specify the hexadecimal code number of the key (\$2E). The MIDI Macro, however, allows for a much easier way: merely input the name of the key with the ' prefix (apostrophe).

Examples:

'C3 (= \$3C)

'A#1 (= \$2E)

Note:

- C3 is the name of middle C (refer to the key note table in Appendix D).
- The first character indicates the note; the number specifies the octave.
- Use # for sharps and b for flats (A#3, Db2).

Input of Character Strings

Whenever the input of a character string is required (voice name of the DX7, for example), use quotation marks.

Example:

"BRASS"

The **A** command will replace each character of the string by its ASCII code.

"BRASS" → 42 52 41 53 53 (hexadecimal)

Using Symbols

One of the most powerful features of the Macro Assembler is its ability to accept symbols instead of a raw sequence of numbers.

There are two kinds of symbols, global and local. Global symbols are defined in the Macro Symbol Edit screen, accessed by the **S** command; local symbols are defined in each individual block of data. The **A** command refers to the Global Symbol Table in order to convert the symbol into numbers whenever it appears. A local symbol, however, will be converted into the contents it was assigned in the block where it appears.

Therefore, you will use global symbols for the sequences of numbers which appear frequently in any of your source blocks; you will use local symbols when you want the same symbol to be assigned different contents in each block.

To define a global symbol, use the **S** command to switch to the Macro Symbol Editor. Type in the name of your symbol, the equal sign, then the value.

Example:

EOX = \$F7

If you want the symbol to be assigned more than one value, type in these values between < and > brackets, separating successive numbers by a comma.

Example:

Sample = <\$F0, \$43, \$00, \$01, \$F7>

To define a local symbol inside a source data block, proceed in the same way. Be sure, however, that the definition of a symbol appears in your text before the first utilization.

Note:

- The length of a symbol can vary. However, only the first four characters are taken into consideration:

AAAAB and AAAAC cannot be differentiated.

However, it is often convenient to use symbols made of more than 4 characters for ease of understanding (example: SAMPLE). Be careful not to define another symbol starting with the same four characters. The available characters are restricted to:

A ~ Z, a ~ z, 0 ~ 9, _ (underscore)

and the first character must be a letter.

- Upper-case and lower-case letters cannot be distinguished.
- A previously defined symbol may appear in the definition of a new symbol.

Using Operators

Operators can be applied to one or two operands. When the **A** command is activated, the operation is carried out, and its result substituted for the expression.

Constants and one-byte symbols are valid operands (applying an operator to a symbol that is assigned more than one byte of data is invalid).

The following table gives a list of the available operators.

Operator Symbol	Operation	Example/Comments
*	Multiplication	\$20*2 → \$40
/	Division	\$10/2 → \$08
+	Addition	\$90 + 6 → \$96
-	Subtraction	68 - 34 → \$22
-	Two's Complement	-\$40 → \$C0 \$40 = % 01000000 ↓ Reverse each bit ↓ % 10111111 ↓ Add 1 (discard any carry over beyond the MSB) ↓ % 11000000 ↓ \$C0
!	Bit Reversion	!\$00 → \$FF
AND	Logical AND between homologous bits	% 1001 AND % 1100 → % 1000
OR	Logical OR between homologous bits	% 1001 OR % 1100 → % 1101
XOR	Logical OR (exclusive) between homologous bits	% 1001 XOR % 1100 → % 0101

★ For more details about logical operators and two's complement, refer to your Basic manuals (what applies in Basic for two-byte operands applies here for one-byte operands).

Note:

Two's complement of a one-byte is used as a consistent representation of a negative value. The reader can easily check the following identity:

$$A - B = A + (-B)$$

where A and B are both one-byte values, and (-B) is the two's complement of B. (Discard any carry over beyond the MSB.)

Using DHL and DLH Functions

• The DHL Function

The **DHL** function splits its argument into two bytes. The argument is written in brackets and its value must fit on 14 bits.

Example:

DHL <160> → \$01, \$20

To understand this result let's write 160 in binary notation, on 14 bits.

160 decimal = 00000010100000 binary, 14 bits

Now, let's split the two halves of the binary notation.

% 0000001	,	% 0100000
↓		↓
\$01	,	\$20
(High 7 bits)		(Low 7 bits)

This function is used to simplify the input of the Byte Count when using Bulk data with Yamaha System Exclusive message. The function allows for easy input of a value higher than 127 to be encoded on 2 bytes of 7 significant bits.

YAMAHA BULK DATA FORMAT

11110000	Status = FO(H)
01000011	ID = 43(H)
0000nnnn	n = Channel Number
0ffffff	f = Format Number
0bbbbbbb	b = Byte Count (Number of Bytes)
0bbbbbbb	
0ddddddd	d = Data
0ddddddd	
0eeeeeee	e = Check Sum
11110111	EOX = F7

Example:

When the Bulk corresponds to voice data, the format number = 00(H), and the number of bytes = 155:

\$FO, \$43, \$00, DHL <155>, data....., data, check sum, \$F7

The check sum is the two's complement of the lower 7 bits of the data sum.

Note:

DHL <value> = byte1, byte2 can be easily related to Basic expressions as follows:

byte1 = value\128 (\ integer division)
 byte2 = value MOD 128

• **The DLH Function**

The DLH function works in the same way as the DHL function but the lower 7 bit byte comes first.

Example:

DHL <\$2000> → \$40, \$00
DLH <\$2000> → \$00, \$40

This function is useful to match the format of the Pitch Bender value. The status byte for the Pitch Bender is \$E0 and is followed by the bytes of data. These two bytes can encode values from \$0000 to \$3FFF and the lower 7 bit bytes must come first.

Note:

Each time you have some doubt about the result of a function or an operation, you may temporarily input the corresponding data in an unused block, assemble this block (**A** command), and then have a look at the object code (**O** command).

Inserting Comments

A source data text that you are creating today may include symbols corresponding to complex operations. If you are to retrieve the same text one month after, you will welcome comments giving detailed explanation of what the text and symbol stand for.

A comment starts with a ; (semi-colon). Any data following a semi-colon, and belonging to the same line, will be considered as a comment and will be discarded by the **A** command.

This is quite similar to the REM statement of Basic.

Operator	Description	Example
OR	Logical OR	When the Bulk corresponds to value data the format (number = DHL) and the number of bytes = 155
XOR	Logical XOR	The check sum is the two's complement of the lower 7 bits of the data sum

* For more details about logical operators and two's complement, see the appendix.

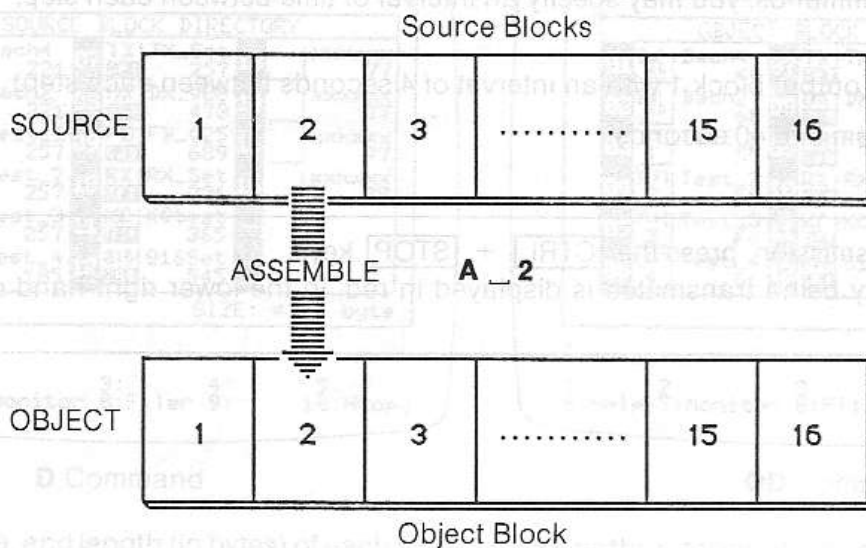
ASSEMBLING & TRANSMITTING THE DATA

Assembling and transmitting were outlined in a previous section of this chapter ("Operation Guideline"). Following, are some details about these operations.

Assembling

When the source data input is complete, switch to the command mode (and press the **F6** key if you want the Command Menu to be displayed).

Type in **A**, a space, the number of the block to be assembled, then press **RETURN**. If you omit the block number, the last edited block will be assembled.



- ★ To assemble all the blocks, use the **AA** command (without parameters).
If assembling can be carried out (no error in the source data), the message "Complete!" appears in the message area. Otherwise, an error message appears in the following format:

error message [b, l] b = block number
 l = line number

For a list of the error messages please refer to the Appendix.

Displaying the Object Data

To display one block of object data, use the **O** command. You may specify the block number, otherwise the last assembled block will be displayed.

MIDI MACRO ASSEMBLER				SRC	OBJ
[BLOCK INFORMATION]				Font	<input type="checkbox"/>
No.	Name	Size			
01	xxxxxxx	149	<input type="checkbox"/>		
1	\$c0,5				
2	\$b0,7,110				
3	\$b0,1,0				
4					
5					
6					
7					
8					
9					
10					
11					
12					

O_1

→

MIDI MACRO ASSEMBLER				SRC	OBJ
[BLOCK INFORMATION]				Font	<input type="checkbox"/>
No.	Name	Size			
01	xxxxxxx	48	<input type="checkbox"/>		
	000	C0 05 B0 07 6E B0 01 00			
	001				
	002				
	003				
	004				
	005				
	006				
	007				
	008				
	009				
	010				
	011				

Transmitting the MIDI Data

Use the **OU**, **Lout**, or **SO** command to send the object data to the MIDI OUT connector. You may specify the block number. When the block number is omitted, the last assembled block will be transmitted.

OU transmits one block once

L transmits one block repetitively, with an interval of about 3 seconds between consecutive transmissions.

SO transmits a single byte each time you press the space bar after entering the command.

With the **OU** and **L** commands, you may specify an interval of time between each step.

Example:

OU _1, 4 (output block 1 with an interval of 4 seconds between each step)

The interval ranges from 0 to 40 seconds.

Note:

- To abort the transmission, press the **CTRL** + **STOP** keys.
- The step currently being transmitted is displayed in red, in the lower right-hand corner of the screen.

Displaying the Object Data

To display the block of object data, use the **O** command. You may specify the block number. Otherwise the last assembled block will be displayed.



UTILITY COMMANDS

The commands listed in the lower part of the Help Menu are utility commands: they provide the tools for re-arranging your blocks. Each of these commands accepts the prefix **O** (object). A command without this prefix will be applied to the source block(s).

Block Directory

The **D** command displays a list of the blocks.

Block	Name	Length	Size
01	Bach4	221	77
02	Bach2	221	77
03	Test_1	257	77
04	Test_2	257	77
05	Test_3	257	77
06	Test_4	185	77

SIZE: 4358 byte

?
1: Help 2: Monitor 3: Filer 4: 5: 6: HCopy

D Command

Block	Name	Length	Size
01	Bach4	48	40
02	Bach2	68	40
03	Test_1	57	40
04	Test_2	58	40
05	Test_3	64	40
06	Test_4	66	40

SIZE: 1439 byte

?
1: Help 2: Monitor 3: Filer 4: 5: 6: HCopy

OD Command

The font, name, and length (in bytes) of each block, as well as the total length of your data are displayed.

Erasing a Block

The **K** command is used to erase a single block. The block number must be specified, otherwise an error message appears. After you enter the command, the corresponding (source or object) directory appears, with the message **Are you sure?** Press **RETURN** to confirm or another key to abort.

Copying a Block

The **C** command allows you to copy one block to another. The first parameter indicates the block to be copied; the second parameter indicates the destination block.

Example:

C _1, 2 copies block 1 on block 2

Important:

The destination block data will be erased and replaced by the data of the block to be copied.

Omitting one parameter will result in an error message.

After you enter the **C** command, the directory appears with the message **Are you sure?** Press **RETURN** to confirm or another key to abort.

Swapping Two Blocks

The **SW** command allows you to exchange two specified blocks. The parameters specifying the two blocks may not be omitted.

Although this operation is harmless (reversible), the directory appears, and you are requested to confirm or cancel the commands.

Printing Blocks

The **P** command allows for the printing of one or more blocks. The parameters specify the first and last block to be printed.

Example:


P_1, 3 prints blocks 1, 2, and 3 and turns the screen background to green during printing.

- ★ To print only one block, enter only one parameter (i.e. **P_1**)
- ★ To print the global symbol table, enter **P_0**.

Important:

To abort the printing operation, press **CTRL** + **STOP**, THEN TURN THE PRINTER OFF AND ON to clear the printer buffer.

Example of printout:

```
SOURCE BLOCK NO.01 [RXInCh] 941 
0001:;==Define Symbol=====
0002:
0003:RXPC=<$F0,$43,$10,3>
0004:
0005:SD1=116
0006: SDHV=0,SDMD=1,SDLI=2,SDHT=3
0007:
0008:BD1=118
0009: BDM1=0,BDM2=1,BDHVY=2
000A:
000B:HHCL=120
000C: HHC1=0,HHC2=1,HHPD=2
000D:
000E:HHOP=121
000F: HH01=0,HH02=1
0010:
0011:EOX=$F7
0012:
0013:;==Inst Change of RX11=====
0014:
0015:RXPC,SD1,SDHV,EOX
0016:RXPC,BD1,BDM1,EOX
0017:RXPC,HHCL,HHC1,EOX
0018:RXPC,HHOP,HH01,EOX
0019:
```

EXAMPLES OF DATA CREATION

As an example, let's try to create data for switching the instrument timbre of a Yamaha RX-11, and the configuration data for the Yamaha FB-01.

RX-11 INST Change

Suppose we want to produce the following settings on an RX-11:

- SD1 → HEAVY
- BD1 → MEDIUM
- HH CLOSE → CLOSED1
- HH OPEN → OPEN1

- **Creating the Source Data**

- First, switch to the Source Data Edit mode, block 1:

E _1 RETURN

The above setting is carried out by a System Exclusive Message having the following formula:

11110000	= F0(H)	Status
01000011	= 43(H)	ID
0001nnnn	= 1n(H)	Sub-status n = MIDI Channel
00000011	= 03(H)	Group and Sub-group Number
0ppppppp	= 0p(H)	Parameter Number
0ddddddd	= 0d(H)	Data
11110111	= F7(H)	EOX

The first byte, F0(H), announces a System Exclusive Message.

The second byte identifies the manufacturer: Yamaha's ID number is 43(H).

The third byte gives the sub-status and the MIDI channel. The sub-status required here is 1.

The group and sub-group numbers indicate the type of data. For an RX-11, this must be 3.

The parameter number indicates the type of parameter.

The Data byte provides the value of the parameter.

The last byte indicates the End Of Exclusive message.

Note that for each message of this type that we have to create, the first four bytes will be identical if they apply to the same MIDI channel. Therefore, it is suitable to define a global symbol for these bytes.

- Switch to the Symbol Edit screen by pressing the F3 key.

Input the following:

RXPC = <\$F0, \$43, \$10, \$03>

RXPC (RX-11 Parameter Change) now stands for the first four bytes of our messages, and will be received on channel 1.

The two next data (Parameter Number and Data) use only one byte each, so we do not need to define a symbol for these. However, defining a symbol for each of these data will make our source text read better.

Type in the following:

SD1 = 116

SDHVVY = 0, SDMD = 1, SDLT = 2, SDHT = 3

BD1 = 118

BDM1 = 0, BDM2 = 1, BDHVVY = 2

HHCL = 120

NNC1 = 0, HHC2 = 1, HHPD = 2
HHOP = 121
HHO1 = 0, HH02 = 1

- ★ The above values are particular to the RX-11.
- ★ We have prepared more values than we actually need for our purpose. This is, however, very convenient for easy modification of the source text, which we may be required to update later.

Now define the last symbol:

EOX = \$F7

This concludes the definition of our symbols.

```

MIDI MACRO ASSEMBLER SRC OBJ
[SYMBOL TABLE INFORMATION]
Size
433
1 RXPC=<$F0,$43,$10,3>
2
3 SD1=116
4 SDHV=0,SDMD=1,SDLI=2,SDHT=3
5 BD1=118
6 BDM1=0,BDM2=1,BDHVY=2
7 HHCL=120
8 HHC1=0,HHC2=1,HHPD=2
9 HHOP=121
10 HHO1=0,HH02=1
11
12 EOX=$F7
  
```

Press the **[ESC]** key to edit block 1. The program will check your definitions at this stage, and if an error is found, an error message will appear and the previous screen will be displayed again.

Note:

You may use an old symbol in the definition of a new symbol. If you have, for instance, already defined a symbol PaCh:

PaCh = <\$F0, \$43, \$10 >

for changing the parameters of a DX 7, you may shorten the above definition and define RXPC as:

RXPC = <PaCh, 3 >

Let's now write our source text. To make SD1 HEAVY, simply enter

RXPC, SD1, SDHVY, EOX

The next input is as follows:

RXPC, BD1, BDM1, EOX
RXPC, HHCL, HHC1, EOX
RXPC, HHOP, HHO1, EOX

```

MIDI MACRO ASSEMBLER SRC OBJ
[BLOCK INFORMATION] Font
No. Name Size
01 RXInCh 185
1 RXPC,SD1,SDHV,EOX
2 RXPC,BD1,BDM1,EOX
3 RXPC,HHCL,HHC1,EOX
4 RXPC,HHOP,HHO1,EOX
5
6
7
8
9
10
11
12
  
```

- Press the **[ESC]** to switch to the Command mode, and assemble the data by entering the **A** command:

A **[RETURN]**

If assembling can be carried out, "Complete!" will appear after a pause; otherwise an error message will indicate in which line of block 1 an error has been found.

- If you want to see the object data, enter the **O** command:

O **[RETURN]**

MIDI MACRO ASSEMBLER										SRC	OBJ
[BLOCK INFORMATION]										Font	
No.	Name	Size								Font	
01	RXInCh	68									
000	F0 43 10 03 74 00 F7 F0 43 10										
001	03 76 00 F7 F0 43 10 03 78 00										
002	F7 F0 43 10 03 79 00 F7										
003											
004											
005											
006											
007											
008											
009											
010											
011											

To check whether this Macro works or not, you need an RX-11 connected to the MIDI OUT of your computer.

OU _1 **[RETURN]**

will send the data, and you can check the actual changes.

- ★ The RX-11 must be set to SYS INFO ON to actually receive data.

Note:

If you want to save this Macro, refer to the next section of this chapter. Such a Macro can be later retrieved from the File mode of the RX Editor (YRM-302).

Configuration for the FB-01

Let's now try to create a configuration of the FB-01.

- Connect the MIDI OUT of the FB-01 to the MIDI IN of the computer, and the MIDI OUT of the computer to the MIDI IN of the FB-01.
- Switch to the MIDI Monitor program, Monitor mode to enable the reception of data from the FB-01 (press **[F7]**).
- Operate the FB-01 to effect a bulk dump. The data will appear on your screen.
- Press **[F5]** to switch to the Memory Dump mode. The data stored in the memory should be between the 0001 and 00AB addresses. The data should consist of F0(H), 43(H), 75(H), 00(H), 00(H), 01(H)F7(H).
- ★ If you already had something in the memory, the start and end addresses will be different.
- Transfer the data into the Paste Buffer.

GE _B0, 1 **[RETURN]**

To check that the data are correctly transferred into the Paste Buffer, copy the same data at a free location of the memory:

PU _200 **[RETURN]**

A — LIST OF THE COMMANDS

Note:

In the following lists of commands, we adopted the following conventions:

- In the "Command" column we give the whole name of the commands. Only upper case character(s) have to actually be input.
- The "Format" column indicates the syntax of the command. Parenthesis and the " _ " mark are metasymbols. They do not have to be input. Parenthesis indicate an optional parameter; " _ " indicates that a blank space is required.
- The effect of the command is described in the "Function" column.
- The default value for optional parameters is given in the same column.
- If not otherwise specified, **m** and **n** stand for block numbers (1 ~ 16).

MIDI Monitor Program

• Commands available in both Monitor and Memory Dump Modes

Command	Format	Function
Clear	CL	Clears the memory
FRee	FR	Returns the number of bytes left for symbol registration
Llist	LI	Displays the list of registered symbols
SClear	SC	Erases all registered symbols
PRinter	PR _ type, mode	Specifies the printer and printer mode type = MSx or EPson mode = Single or DOuble

• Commands Available in Memory Dump Mode Only

Command	Format	Function
GEt	GE _ m, n	transfers the data stored in the memory between the addresses m and n into the Paste Buffer. m = start address n = end address m < n
PUt	PU _ m	Copies the contents of the Paste Buffer into the memory from the address m .

MIDI Macro Assembler Program

Command	Format	Function
AAssemble	AA	Converts all source data blocks into object data blocks.
Assemble	A_(m)	Converts source data block m into object data block m . Default for m : the last edited block.
Copy	C_m, n	Copies the source data block m into the source data block n .
Directory	D	Displays a table of the source data blocks (name, font, length).
Edit	E_(m)	Allows for editing of the source data block m . Default for m : the last edited block (or 1, when starting the editing)
Kill	K_m	Deletes the source data in block m .
Lout	L_(m) (,t)	Periodic output of the object data block m . t is the interval (in seconds) between the output of two consecutive bytes. Default for m : the last assembled block Default for t : 0
Object	O_(m)	Displays the object data block m . Default for m : the last assembled block
Out	OU_(m) (,t)	Output of the object data block m . t is the interval (in seconds) between output of two consecutive bytes. Default for m : the last assembled block Default for t : 0
ODirectory	OD	Displays a table of the output blocks (name, font, length).
OCopy	OC_m, n	Copies the object data block m into the object data block n .
OKill	OK_m	Deletes the object data in block m .
OSW	OSW_m, n	Swaps object data in blocks m and n .
OPrint	OP_m(n)	Prints out the object data blocks from m to n . Default for n : m
Print	P_m(n) P_0	Prints out the source data blocks from m to n . Default for n : m Prints out the list of global symbols.
SWap	SW_m, n	Swaps source data in blocks m and n .
SOut	SO_(m)	Set the object data block m to single stop output. A byte is transmitted by pressing the space bar. Default for m : the last assembled block
Sedit	S	Switches to Macro Symbol Editor screen.

B — KEYBOARD OPERATIONS

MIDI Monitor Program

• Monitor Mode

Category	Key	Function
Execution and Function Switching	[F1]	Hexadecimal/ASCII
	[F2]	Line Feed ON/OFF
	[F3]	Display Filter ON/OFF
	[F4]	Merge Filter ON/OFF
	[F9]	Screen display hold ON/OFF
	[F10]	Hard Copy
	[SELECT]	MIDI Merge ON/OFF Clears the memory
	[CODE]	Printer Dump ON/OFF
	[CTRL] + [STOP]	Interrupts print out
	[CTRL] + [Z]	Key click ON/OFF
Edit	[←] [→]	Cursor movement
	[TAB]	Writes the contents of the template
	[INS]	Insert mode ON/OFF
	[DEL]	Character delete (cursor stationary)
	[BS]	Back space and character delete
	[RETURN]	Enters a command
	[ESC]	Cancels a command input
Help	[F6]	Displays the Help screen
	[RETURN]	Next page
	[ESC]	Back to Monitor mode
Switching to another mode	[F5]	Memory Dump mode
	[F7]	MIDI Macro Assembler program
	[F8]	Filter Edit mode

• Memory Dump Mode

Category	Key	Function
Scrolling	↑ ↓	Scrolling (one line)
	F2	Scrolling (to previous block)
	F3	Scrolling (to next block)
Execution and Function Switching	F1	Hexadecimal/ASCII
	F4	Output Filter ON/OFF
	F10	Hard Copy
	CTRL + STOP	Interrupt printout
Edit	← →	Cursor movement
	TAB	Writes the contents of the template
	INS	Insert mode ON/OFF
	DEL	Character delete (cursor stationary)
	BS	Back space and character delete
	RETURN	Enters a command
	ESC	Cancels command input
Help	F6	Displays the Helpscreen
	RETURN	Next page
	ESC	Back to Memory Dump mode
Switching to another mode	F5	Monitor mode
	F7	MIDI Macro Assembler program
	F8	Filter Edit mode

• Filter Edit Mode

Category	Key	Function
Edit	↑ ↓ ← →	Cursor movement
	RETURN	Selected data ON/OFF
	SPACE	Same as RETURN
	F1 / F6	Message ON/OFF
	F2 / F7	Channel ON/OFF
	F3 / F8	System Messages as a whole ON/OFF
Execution	F5	Recalls previous setting
	F10	Hard copy
	CTRL + STOP	Interrupts print out
Exit	ESC	Back to Monitor or Memory Dump mode

MIDI Macro Assembler Program

• **Command Mode**

Category	Key	Function
Edit	← →	Cursor movement
	TAB	Writes the contents of the template
	INS	Insert mode ON/OFF
	DEL	Character delete (cursor stationary)
	BS	Back Space and character delete
	RETURN	Enters a command
	ESC	Cancels a command input
Execution	F10	Hard copy
	CTRL + STOP	Interrupt print out
Help	F6	Displays Help screen
Switching to another mode	F7	MIDI Monitor program
	F8	File mode

• **File Mode**

Category	Key	Function
Specification and execution	↑ ↓ ← →	Cursor movement
	RETURN	Selects an item/enters a file name
	ESC	Back to Command mode

• Edit Mode

Category	Key	Function
Edit	↑ ↓ ← →	Cursor movement
	SHIFT + ↑ ↓	Cursor movement (one page)
	SHIFT + ← →	Cursor movement (one block)
	BS	Back space and character delete
	DEL	Character delete (cursor stationary)
	SHIFT + DEL	Deletes one line
	CTRL + E	Deletes all the characters to the right of the cursor
	INS	Insert mode ON/OFF
	SHIFT + INS	Creates a blank line below the cursor
	RETURN	Line break
	F4	Stores the line the cursor is on
	F5	Writes the contents of the Paste Buffer at the cursor position (source data editing)
F9	Writes the line stored by F4	
Registration	F1	Block name
	F2	Block font
	F3	Switches to the Symbol Edit screen
Execution	F10	Hard copy
	CTRL + STOP	Interrupts print out
Exit	ESC	Back to Command mode
Font Edit	↑ ↓ ← →	Cursor movement
	SPACE	Blue/White dot
	DEL	Reflection
	INS	Color reverse
	HOME	90° rotation (counterclockwise)
	SHIFT + ↑ ↓	Periodic translation (32 x 32 dots)
	← →	Periodic translation (32 x 32 dots)
	RETURN	Enters the font
ESC	Cancels the font registration	

C — ERROR MESSAGES

MIDI Monitor Program

Message	Cause	Remedy
Illegal character	Illegal characters were used	Use legal characters only
Illegal value	Data value is incorrect	Use a value in the valid range
Undefined symbol	Undefined symbols were used	Use only defined symbols
Illegal operand	Incorrect command parameter was used	Input parameters correctly
Can't transmit it now	Data output is not possible in the current mode	Use the Monitor mode
Illegal command name	Incorrect command name was used	Use correct command names
Illegal symbol name	Incorrect symbol name was used	Use correct symbol names
Symbol table overflow	The symbol table is full	Delete unnecessary symbols
Too long!!	A symbol is too long	A symbol should be shorter than 255 bytes

MIDI Macro Assembler Program

Message	Cause	Remedy
Illegal command	Incorrect command was used	Use correct commands
Illegal argument	Incorrect command parameters were used	Input correct parameters
SOURCE area is full	The source area is full	Delete unnecessary source data
This block is empty	Trying to output an empty block	First create object data, then output

Assembling Errors

Message	Cause	Remedy
Illegal character	Illegal character used	Use only legal characters
Syntax error	Format error	Correct the error
Type mismatch	Different type of data was operated	Correct
Undefined symbol	Undefined symbol used	Use only defined symbols
Data overflow	The result of operation exceeds the range (3FFFh)	Shorten the data
Memory is full	The memory area is full	Erase unnecessary data
Multiple definition	The symbol is defined twice	Delete one of the definitions
Division by zero	A division by zero was attempted	Correct the denominator
Undefined function	Non-usable function was used	Refer to Chapter 3, and use only valid functions and operators
OBJECT area is full	The OBJECT area became full	Kill unnecessary blocks

File Mode

Message	Cause	Remedy
Read Error	An error has occurred during data loading	Check the state of the external device
Write Error	An error has occurred during data saving	Check the state of the external device
Write Protected	The floppy disk is write-protected	Slide the write-protection tab to the closed position
Illegal File Name	Illegal file name used	Specify a correct file name
File not Found	The file specified was not found	Specify the correct file name (display the directory)
Disk not Ready	The floppy disk is not correctly set up	Set the disk correctly
Disk Full	Trying to save data to a floppy disk with no available area	Delete unnecessary file or use a new disk
Illegal Data Type	Due to the different file type, loading is not enabled	This occurs with tapes because a file name extension cannot be used. Try to make it a habit to write the list of file names on the tape directory.
Verify Error	An error has occurred while verification was carried out by the data recorder.	Try saving again. Adjust the volume, tone, and phase controls. Clean the magnetic head.

D — MIDI KEY CODE (FOR PITCH SPECIFICATION)

Key Name	Code		Key Name	Code		Key Name	Code		Key Name	Code	
	Dec.	Hex.		Dec.	Hex.		Dec.	Hex.		Dec.	Hex.
C-2	0	0	C1	36	24	C4	72	48	C7	108	6C
C#-2	1	1	C#1	37	25	C#4	73	49	C#7	109	6D
D-2	2	2	D1	38	26	D4	74	4A	D7	110	6E
D#-2	3	3	D#1	39	27	D#4	75	4B	D#7	111	6F
E-2	4	4	E1	40	28	E4	76	4C	E7	112	70
F-2	5	5	F1	41	29	F4	77	4D	F7	113	71
F#-2	6	6	F#1	42	2A	F#4	78	4E	F#7	114	72
G-2	7	7	G1	43	2B	G4	79	4F	G7	115	73
G#-2	8	8	G#1	44	2C	G#4	80	50	G#7	116	74
A-2	9	9	A1	45	2D	A4	81	51	A7	117	75
A#-2	10	A	A#1	46	2E	A#4	82	52	A#7	118	76
B-2	11	B	B1	47	2F	B4	83	53	B7	119	77
C-1	12	C	C2	48	30	C5	84	54	C8	120	78
C#-1	13	D	C#2	49	31	C#5	85	55	C#8	121	79
D-1	14	E	D2	50	32	D5	86	56	D8	122	7A
D#-1	15	F	D#2	51	33	D#5	87	57	D#8	123	7B
E-1	16	10	E2	52	34	E5	88	58	E8	124	7C
F-1	17	11	F2	53	35	F5	89	59	F8	125	7D
F#-1	18	12	F#2	54	36	F#5	90	5A	F#8	126	7E
G-1	19	13	G2	55	37	G5	91	5B	G8	127	7F
G#-1	20	14	G#2	56	38	G#5	92	5C			
A-1	21	15	A2	57	39	A5	93	5D			
A#-1	22	16	A#2	58	3A	A#5	94	5E			
B-1	23	17	B2	59	3B	B5	95	5F			
C0	24	18	C3	60	3C	C6	96	60			
C#0	25	19	C#3	61	3D	C#6	97	61			
D0	26	1A	D3	62	3E	D6	98	62			
D#0	27	1B	D#3	63	3F	D#6	99	63			
E0	28	1C	E3	64	40	E6	100	64			
F0	29	1D	F3	65	41	F6	101	65			
F#0	30	1E	F#3	66	42	F#6	102	66			
G0	31	1F	G3	67	43	G6	103	67			
G#0	32	20	G#3	68	44	G#6	104	68			
A0	33	21	A3	69	45	A6	105	69			
A#0	34	22	A#3	70	46	A#6	106	6A			
B0	35	23	B3	71	47	B6	107	6B			

E — CONVERSION TABLE

Dec.	Bin.	Hex.	Dec.	Bin.	Hex.	Dec.	Bin.	Hex.	Dec.	Bin.	Hex.
0	00000000	0	64	01000000	40	128	10000000	80	192	11000000	C0
1	00000001	1	65	01000001	41	129	10000001	81	193	11000001	C1
2	00000010	2	66	01000010	42	130	10000010	82	194	11000010	C2
3	00000011	3	67	01000011	43	131	10000011	83	195	11000011	C3
4	00000100	4	68	01000100	44	132	10000100	84	196	11000100	C4
5	00000101	5	69	01000101	45	133	10000101	85	197	11000101	C5
6	00000110	6	70	01000110	46	134	10000110	86	198	11000110	C6
7	00000111	7	71	01000111	47	135	10000111	87	199	11000111	C7
8	00001000	8	72	01001000	48	136	10001000	88	200	11001000	C8
9	00001001	9	73	01001001	49	137	10001001	89	201	11001001	C9
10	00001010	A	74	01001010	4A	138	10001010	8A	202	11001010	CA
11	00001011	B	75	01001011	4B	139	10001011	8B	203	11001011	CB
12	00001100	C	76	01001100	4C	140	10001100	8C	204	11001100	CC
13	00001101	D	77	01001101	4D	141	10001101	8D	205	11001101	CD
14	00001110	E	78	01001110	4E	142	10001110	8E	206	11001110	CE
15	00001111	F	79	01001111	4F	143	10001111	8F	207	11001111	CF
16	00010000	10	80	01010000	50	144	10010000	90	208	11010000	D0
17	00010001	11	81	01010001	51	145	10010001	91	209	11010001	D1
18	00010010	12	82	01010010	52	146	10010010	92	210	11010010	D2
19	00010011	13	83	01010011	53	147	10010011	93	211	11010011	D3
20	00010100	14	84	01010100	54	148	10010100	94	212	11010100	D4
21	00010101	15	85	01010101	55	149	10010101	95	213	11010101	D5
22	00010110	16	86	01010110	56	150	10010110	96	214	11010110	D6
23	00010111	17	87	01010111	57	151	10010111	97	215	11010111	D7
24	00011000	18	88	01011000	58	152	10011000	98	216	11011000	D8
25	00011001	19	89	01011001	59	153	10011001	99	217	11011001	D9
26	00011010	1A	90	01011010	5A	154	10011010	9A	218	11011010	DA
27	00011011	1B	91	01011011	5B	155	10011011	9B	219	11011011	DB
28	00011100	1C	92	01011100	5C	156	10011100	9C	220	11011100	DC
29	00011101	1D	93	01011101	5D	157	10011101	9D	221	11011101	DD
30	00011110	1E	94	01011110	5E	158	10011110	9E	222	11011110	DE
31	00011111	1F	95	01011111	5F	159	10011111	9F	223	11011111	DF
32	00100000	20	96	01100000	60	160	10100000	A0	224	11100000	E0
33	00100001	21	97	01100001	61	161	10100001	A1	225	11100001	E1
34	00100010	22	98	01100010	62	162	10100010	A2	226	11100010	E2
35	00100011	23	99	01100011	63	163	10100011	A3	227	11100011	E3
36	00100100	24	100	01100100	64	164	10100100	A4	228	11100100	E4
37	00100101	25	101	01100101	65	165	10100101	A5	229	11100101	E5
38	00100110	26	102	01100110	66	166	10100110	A6	230	11100110	E6
39	00100111	27	103	01100111	67	167	10100111	A7	231	11100111	E7
40	00101000	28	104	01101000	68	168	10101000	A8	232	11101000	E8
41	00101001	29	105	01101001	69	169	10101001	A9	233	11101001	E9
42	00101010	2A	106	01101010	6A	170	10101010	AA	234	11101010	EA
43	00101011	2B	107	01101011	6B	171	10101011	AB	235	11101011	EB
44	00101100	2C	108	01101100	6C	172	10101100	AC	236	11101100	EC
45	00101101	2D	109	01101101	6D	173	10101101	AD	237	11101101	ED
46	00101110	2E	110	01101110	6E	174	10101110	AE	238	11101110	EE
47	00101111	2F	111	01101111	6F	175	10101111	AF	239	11101111	EF
48	00110000	30	112	01110000	70	176	10110000	B0	240	11110000	F0
49	00110001	31	113	01110001	71	177	10110001	B1	241	11110001	F1
50	00110010	32	114	01110010	72	178	10110010	B2	242	11110010	F2
51	00110011	33	115	01110011	73	179	10110011	B3	243	11110011	F3
52	00110100	34	116	01110100	74	180	10110100	B4	244	11110100	F4
53	00110101	35	117	01110101	75	181	10110101	B5	245	11110101	F5
54	00110110	36	118	01110110	76	182	10110110	B6	246	11110110	F6
55	00110111	37	119	01110111	77	183	10110111	B7	247	11110111	F7
56	00111000	38	120	01111000	78	184	10111000	B8	248	11111000	F8
57	00111001	39	121	01111001	79	185	10111001	B9	249	11111001	F9
58	00111010	3A	122	01111010	7A	186	10111010	BA	250	11111010	FA
59	00111011	3B	123	01111011	7B	187	10111011	BB	251	11111011	FB
60	00111100	3C	124	01111100	7C	188	10111100	BC	252	11111100	FC
61	00111101	3D	125	01111101	7D	189	10111101	BD	253	11111101	FD
62	00111110	3E	126	01111110	7E	190	10111110	BE	254	11111110	FE
63	00111111	3F	127	01111111	7F	191	10111111	BF	255	11111111	FF

F — SUMMARY OF MIDI

MIDI (Musical Instrument Digital Interface) is a world wide standard adopted for the communication between musical instruments and computers. This standard specifies what kind of elementary information can be exchanged, and in which electronic format this information must be shaped. Elementary information is called a MIDI Message. A MIDI Message is itself divided into smaller pieces of information called bytes. A byte is a succession of eight bits. A bit can take two values (0 or 1), which correspond to two digital voltages when set to travel through a MIDI cable or to be processed by a digital interface. Therefore, a byte can have 256 bit configurations, and each of these configurations can be associated with a number from 0 to 255. A byte will therefore read as an eight digit binary number.

According to this, you may see the MIDI information as a sequence of numbers, each one in the range of 0 to 255. And a MIDI message will correspond to one, two, or more of such numbers.

The MIDI standard specifies the structure of the information by defining a set of MIDI messages and the structure of each message. The MIDI message also concerns some hardware specifications. For instance, the MIDI information must travel through a single cable. When more than one instrument are to receive the information from a single source, the instruments are daisy-chained, that is, the same information reaches all slave instruments and each instrument has to pick out only the portion of the MIDI message that is relevant to them. This is possible because a message can carry an identifier called a MIDI Channel Number. The MIDI Standard allows for 16 different MIDI Channels. This works as follows: if a MIDI instrument is set to receive on MIDI Channel 1, for example, a message carrying a different MIDI Channel Number will be discarded. This feature allows for the transmission of specific information to 16 different instruments through a single cable.

Structure of the MIDI Information

There are two types of MIDI Messages.

- **The Channel Messages**

The Channel Messages carry a MIDI Channel Number so that they cannot be received by any instruments set to receive another channel.

- **The System Messages**

The System Messages will be, in principle, received by all instruments, regardless of their channel setting. The System Common Messages and the System Real Time Messages can be received by every MIDI instrument; the System Exclusive Messages, however, are individual to different manufacturers and can only be received by instruments of the same brand.

Exceptions:

- In some instances a MIDI Channel Message may be discarded by an instrument, not because the channel numbers do not match, but due to the hardware limitation of that instrument. Example: pitch out of range.
- As we will see later, some System Messages are received only by instruments of a certain brand and will be discarded by other instruments. Further, some System Messages may carry Channel Number information.

Structure of a MIDI Message

A MIDI Message is made of one, two, or more bytes. The first byte is called the status byte, while the next bytes of the same message are called the data bytes.

• Status Byte

The status byte has two functions: identifying the message and carrying the MIDI Channel Number (in the case of a Channel Message). The MIDI Channel Number is encoded on the four lower bits of the status byte.

• Data Bytes

The data bytes carry values relevant to the information specified by the status byte. Example: a NOTE ON message contains the pitch value of the note to be played, and its velocity (volume).

In order for the status byte to be properly interpreted, we need a system to allow the receiving instrument to distinguish which bytes are status bytes and which are data bytes. The trick is, the MSB of a status byte is always 1, while the MSB of a data byte is always 0. Therefore, status bytes range from 80(H) to FF(H), and data bytes from 00(H) to 7F(H).

	Hex.	Dec.	Bin.
Status	80 ~ FF	128 ~ 255	10000000 ~ 11111111
Data	0 ~ 7F	0 ~ 127	00000000 ~ 01111111

We may think on one hand, that we do not need as much as 128 different status bytes, and on the other hand, that 128 different values is too few for some parameters like Pitch Bend.

Well, we do not actually have as many as 128 different status bytes, because, for Channel Messages, the lower four bits are used for Channel Number specifications. It is true, however, that some status bytes are not used - they are kept for future extensions. For parameters requiring a wide range of values, we can split the values on two data bytes, and obtain a range from 0 to 16383.

MIDI Message Format

	Message	Status Byte	First Data Byte (xx)	Second Data Byte (yy)		
	Note Off	8n	Note Number	Velocity		
	Note On	9n	"	"		
	Polyphonic Aftertouch	An	"	Pressure		
CHANNEL MESSAGES	Control Change	Bn	(Control Number) 01 Modulation Wheel 02 Breath Controller 04 Foot Controller 05 Portamento Time 06 Data Entry Slider 07 Main Volume 40 Sustain 41 Portamento 42 Sostenuto 43 Soft 60 Data Increment 61 Data Decrement 7A Local 7B All Note Off 7C Omni Off 7D Omni On 7E Mono On 7F Poly On	Data " " " " " " " 00: Off 7F: On 7F 7F 00: Off, 7E: On 00 00 00 00-0A(Number of Channels) 00		
	Program Change	Cn	Program Number			
	Channel Aftertouch	Dn	Pressure			
	Pitch Wheel	En	Lower 7 bits of the setting	Higher 7 bits of the setting		
	SYSTEM MESSAGES	COMMON MESSAGES	System Exclusive	F0	Manufacturer ID code	Up to the manufacturer
				F1		
			Song Position Pointer	F2	Lower 7 bits of the setting	Higher 7 bits of the setting
			Song Select	F3	Song Number	
				F4, F5		
			Tune Request	F6		
			End of Exclusive	F7		
		REAL TIME MESSAGES	Timing Clock	F8		
			F9			
Start			FA			
Continue			FB			
Stop			FC			
			FD			
Active Sensing			FE			
System Reset	FF					

★ All numbers in the above table are given in Hexadecimal notation.

- **8n(H) NOTE OFF**

1000nnnn The note number indicates which key was released, and velocity indicates how quickly it was released. Very few keyboards have release Velocity Sensitivity (the Sequential Circuits Prophet T8 is one). Most other keyboards (such as the Yamaha DX series) send a Note On message with a velocity of 0 to indicate a Note Off.
n = Channel #

- **9n(H) NOTE ON**

1001nnnn The note number indicates which key was pressed, and velocity indicates how hard it was hit. On keyboards that do not have velocity sensitivity (such as the DX21), a medium value of 40(H) is sent. A Note On message with a velocity of 0 is the same as a Note Off message.
n = Channel #

- **An(H) POLYPHONIC AFTERTOUCH**

1010nnnn The note number indicates which key is being pressed, and the pressure indicates how hard that key is being pressed. (ie. each key can send independent aftertouch messages.) Of all Yamaha keyboards, only the DX1 is able to send (and react to) this message.
n = Channel #

- **Bn(H) CONTROL CHANGE**

1011nnnn The control number indicates which control number is being moved, and the data indicates the position of the controller. In the above chart, control changes 1 ~ 7 are "continuous controllers". (Slider or wheel-type controllers) They carry data in the range of 00(H) ~ 7F(H).
n = Channel #
Control changes 40(H) ~ 43(H) are on/off switch-type controllers, and carry data of either 00(H) or 7F(H).
Control changes 7A(H) ~ 7F(H) are a special type of control change called Mode Messages, and usually carry a fixed data byte. They tell the receiving tone generator how to behave. The way in which these messages are interpreted will depend on the device. (See the MIDI Implementation Chart for your tone generator or synthesizer.)

- **Cn(H) PROGRAM CHANGE**

1100nnnn This tells the receiving device to switch programs (voice memories).
n = Channel #

- **Dn(H) CHANNEL AFTERTOUCH**

1101nnnn Also called "Common Aftertouch", this is found on the DX7.
n = Channel #

- **En(H) PITCH WHEEL**

1110nnnn To provide finer resolution, this data is sent in two bytes, the lower byte first. Yamaha tone generators and synthesizers ignore the higher (second) byte.
n = Channel #

• F0(H) SYSTEM EXCLUSIVE

11110000

After F0(H) must come an identification number which has been assigned to each manufacturer. Yamaha's number is 43(H). What comes between this message and F7(H) (End of Exclusive) is completely up to each manufacturer (but each byte must be between 00(H) and 7F(H)). The data may include a Channel ?, for example. Yamaha uses system Exclusive messages to transmit voice data, sequence data, rhythm pattern data, bulk memory data of all kinds, and many other useful things. See the System Exclusive format chart for your device.

• F7(H) END OF EXCLUSIVE

11110111

This marks the end of a System Exclusive message.

• F2(H), F3(H), F8(H), FA(H), FB(H), FC(H), FF(H)

(Song Position Pointer, Song Select, Timing Clock, Start, Stop, Continue, System Reset) are all for controlling sequencers and rhythm machines. See the MIDI Implementation Chart for your device.

• FE(H) ACTIVE SENSING

If there are no MIDI messages that have been sent, one of these is sent every 300msec, just to let the receiving devices know that there is still someone out there. If there have not been any MIDI messages for a long time (like 1/2 a second), the receiving device assumes that some error has occurred (eg. a MIDI cable was pulled out by mistake), and will stop all notes.

• F1(H), F4(H), F5(H), F9(H), FD(H)

These are unused, and reserved for future expansion.

Note:

When the same message has to be sent repetitively, the status byte being the same, it is possible to shorten the sequence and write the status byte only once. Such a status byte is called a **RUNNING STATUS**. This procedure is allowed for Channel messages only.

Example:

```
90 3C 40 90 40 40 90 3C 00 90 40 00
                        ↓
90 3C 40 40 40 3C 00 40 00
```

You may use this procedure when entering data from the keyboard, in Monitor mode.