“Maple Bus 1.0”
Peripheral Hardware Specifications

92key Keyboard for JPN

Revision 0.85

Created by
Sega Enterprises, LTD.
CS Development and Production Dep't, Second Division

Revision :

0.75	1998/4/20	Initial Version
0.80	1998/6/15	Function key explanation revised
		Conditions for optimization revised
		Language type (hardware setting) added
		Explanation of boomerang processing revised
		Note added to list of key codes
		Capacity of Free Device Status modified
0.82	1998/7/10	Explanation of Free Device Status revised
0.83	1998/8/25	The key code's two GUI keys and the APPLICATION key were changed to S1-S3 keys.
0.84	1998/9/17	8th data for device ID modified
		Function elements GUI keys and application keys changed to application keys S1 - S3
		Details for the constituent elements modified as application keys
		GUI for modification key bits changed to S1 and S2
		Some notes for key codes revised, deleted
		Hexidecimal value for Maximum consumption current changed from 2Ch to 90h
0.85	1998/11/16	Preamble to operation summery deleted

* When reading the newest revision and the previous revision, note that additions are indicated by and deletions by —.

Contents

1 Keyboard overview

1.1 Feature

The keyboard device provides an input/output type man-machine interface for input and editing of words, text, number, and symbol, as well as for screen operations and other controls. This peripheral must meet requirements set forth in "Maple-Bus Standard Specifications," with function type being FT6.

1.2 Operations

~~The following operations are planned currently:~~

Key scan

Constantly perform key scanning and keep the latest key data for returning to the host when requested. By doing so, it can respond quickly to data request from the host. All keys are read simultaneously, with not key scanning priority.

Optimization and conditions

a) When no keys are pressed, simultaneous pressing of up to 3 keys can be detected.

b) When 3 or more keys are already pressed, up to 6 keys in total can be detected, provided the combination is not one that results in a boomerang effect. If boomerang does result, an error code is sent for each additional key pressed.

c) When 6 keys are already pressed under the condition given for b) above, the keyboard function will return error code when the 7th key is pressed.

d) If the boomerang effect occurs when three~~two~~ or more keys are pressed simultaneously, then keyboard function will return the error code.

e) When two or more keys are already press and then 2 or more additional keys are pressed simultaneously, the keyboard function sends an error code even for key combinations that do not result in boomerang.

Restrictions

Key repeat (pressing the same key repeatedly to send out the same key code continuously) cannot be set up at keyboard function.

Key repeat, in terms of repeat rate and wait time before the first repeat, can be set by software on the host.

1.3 Keyboard language and keyboard type

The keyboard for this specification is the Japanese 92-key keyboard.

1.4 Elements of keyboard function

Keyboard function is comprised of the following elements:

- Character keys : A-Z,0-9
- Symbol keys : - ^ \ @ [ ] : ; . / \ ! ” # $ % & ’ ( ) = ~ | ` { + * } < > ? _ ,
- Function keys : F1-F12
- Editing keys : Tab, Backspace, Space, CapsLock, Insert, Delete, Home, End, PageUp, PageDown, PrintScreen, ScrollLock, pause, Enter, Esc, BackSpace, Hankaku/Zenkaku, Muhenkan, Zenkouhou Henkan (Tsugikouho), Katakana Hiragana

- Cursor keys : ← ↑ → ↓
- Control keys : Ctrl, Alt, Shift
- ~~Windows environment key~~Application keys : ~~GUI~~S1,S2,S3
- ~~Application key~~
- Error code : No Operation, Rollover Error, POST Fail, Undefined Error

1.5 Details of keyboard function elements

Each element of keyboard function is explained here in details.

Character and symbol keys:A-Z, -,^,\,@,[,],:,;,.,/,\,!,”,#,$,%,&,’,(,),=,~,|,`,{,+,*,},<,>,?,_,,

Each digital key has an ON/OFF value depending on whether it is pressed or released, respectively. When a key is pressed (ON), its key code will be output. The screen shows the character (A-Z), number (0-9) or symbol. If a mark such as `{ + * } is printed on top left of the key's surface, then the Shift key must be held down while pressing the key.

Function keys:F1-F12~~n(n depends on the keyboard type)~~

Each digital key has an ON/OFF value depending on whether it is pressed or released, respectively. When a key is pressed (ON), its key code will be output. The function assigned to the key (e.g. help) is invoked.

Editing keys:Tab, Backspace, Space, CapsLock, Insert, Delete, Home, End, PageUp, PageDown, PrintScreen, ScrollLock, pause, Enter, Esc, ,BackSpace, Hankaku/Zenkaku, Muhenkan, Zenkouhou Henkan (Tsugikouho), Katakana Hiragana

Each digital key has an ON/OFF value depending on whether it is pressed or released, respectively. When a key is pressed (ON), its key code will be output. These are keys to help with editing when using the character and symbol keys. Some typical functions are listed in figure 1.5.1.

Key	Function Example
Space	Add a space. Can also be used as the Henkan key.
Tab	Move the cursor to a tab stop that has been set up, for example, in a wordprocessing software.
CapsLock	Fix letter input (A-Z) to uppercase (similar to holding down the Shift key all the time).
Insert	Toggle between the insert and overwrite modes for use in, for example, wordprocessing software.
Delete	Delete the next character without moving the cursor position.
Home	Move the cursor to its home position.
End	Move the cursor to the end of the line.
PageUp	Scroll the screen up by one page, as used in wordprocessing software, for example.
PageDown	Scroll the screen down by one page, as used in wordprocessing software, for example.
PrintScreen	Save the current screen as image data in the buffer.
ScrollLock	Function as a fixed key. For example, it can allow scrolling of screen with the cursor position fixed.
Pause	Cancel the selected command.
Enter	Confirm the character entered, enter a command, or add next line.
Esc	Cancel the entered command, character before confirmation, etc.
BackSpace	Move back while deleting one character.
Hankaku/Zenkaku	Convert entered character to Hankaku or Zenkaku. Can also be used when entering the Kanji mode.
Muhenkan	Fixed so that no Henkan (conversion) will be carried on the entered character.
Zenkouho Henkan (Tsugikouho Henkan)	Display the previous or next choice for Kana Kanji Henkan(conversion).
Katakana Hiragana	Change input from Katakana to Hiragana in wordprocessing software, etc.

Figure 1.5.1 Example of editing key functions

Cursor keys: ← ↑ → ↓

Each digital key has an ON/OFF value depending on whether it is pressed or released, respectively. When a key is pressed (ON), its key code will be output. The cursor keys move the cursor.

Control keys:Ctrl, Alt, Shift

Each digital key has an ON/OFF value depending on whether it is pressed or released, respectively. When a key is pressed (ON), its key code will be output. The control keys work as illustrated in figure 1.5.2.

Control key	Function
Ctrl	Used in combination with other keys (e.g. as shortcut key)
Alt	Mainly used as menu key. Can also be used to provide shortcut when combined with other keys.
Shift	For toggling between upper and lower case of character input, and for entering symbols.

Figure 1.5.2 Example on functions of control keys

~~Windows environment keys~~Application keys

The digital key has an ON/OFF value depending on whether it is pressed or released, respectively. When a key is pressed (ON), its key code will be output. ~~They can be used to, for example, display the Start menu of the operating system.~~

~~Application keys~~

~~The digital key has an ON/OFF value depending on whether it is pressed or released, respectively. When a key is pressed (ON), its key code will be output. They can be used to, for example, display the popup menu.~~

Error code

It is provided for handling keyboard error. Physically no key is pressed. The error code is returned to the host when seven or more keys are pressed simultaneously, or when the boomerang effect occurs.

2 Circuitry

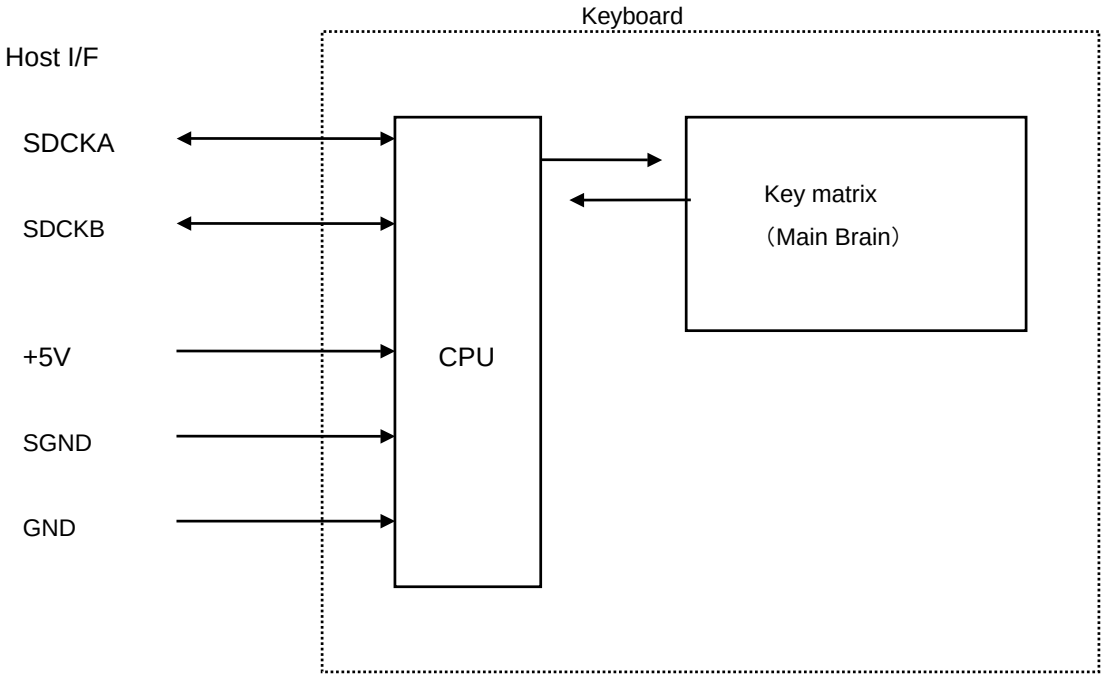


Figure 2 Circuitry

2.1 Language type (hardware setting)

The language type is determined by pulling up or down pins 32 and 34 to 36 (by changing the settings of jumpers J1 to J4 on the board) of the Matsushita CPU specified in the keyboard specifications. The settings are shown in Figure 2.1.

In the figure, the codes shown under "Code" are values that are used for setting the keyboard language (KL) of the device ID set forth in the FT6 specifications. The value 00h is prohibited, so care must be taken to ensure that the values set start with 01h.

Example: When the jumper settings are '0000' , device ID →KL =01h. (In this case, the keyboard language is Japanese.)

Note: In the figure, 1 indicates "pull up," and 0 indicates "pull down."

Language type	ID22 (pin 32)	PO0 (pin 34)	PO1 (pin 35)	PO2 (pin 36)	Code
Japanese	0	0	0	0	01h
English (US)	0	0	0	1	02h
English (UK)	0	0	1	0	03h
German	0	0	1	1	04h
French	0	1	0	0	05h
Italian	0	1	0	1	06h
Spanish	0	1	1	0	07h
Swedish	0	1	1	1	08h
Swiss	1	0	0	0	09h
Dutch	1	0	0	1	0Ah
Portuguese	1	0	1	0	0Bh
Latin America	1	0	1	1	0Ch
Canadian French	1	1	0	0	0Dh
Russian	1	1	0	1	0Eh
Chinese	1	1	1	0	0Fh
Korean	1	1	1	1	10h

Figure 2.1 Language types (hardware settings)

Since the language type of the keyboard described in these specifications is Japanese, all four pins are pulled down (0000).

3 Device ID

The device ID must meet the requirements set forth in "Maple Bus 1.0 Standard Specifications."
The following table indicates the memory image at the host.

3.1 Japanese 92-key keyboard device ID configuration

The device ID is a 16-byte (128 bits) data.

bit	7	6	5	4	3	2	1	0
1st Data	0	0	0	0	0	0	0	0
2nd Data	0	0	0	0	0	0	0	0
3rd Data	0	0	0	0	0	0	0	0
4th Data	0	1	0	0	0	0	0	0
5th Data	0	0	0	0	0	0	0	1
6th Data	0	0	0	0	0	0	1	0
7th Data	0	0	0	0	0	0	0	0
8th Data	~~0~~1	0	0	0	0	0	0	0
9th Data	0	0	0	0	0	0	0	0
10th Data	0	0	0	0	0	0	0	0
11th Data	0	0	0	0	0	0	0	0
12th Data	0	0	0	0	0	0	0	0
13th Data	0	0	0	0	0	0	0	0
14th Data	0	0	0	0	0	0	0	0
15th Data	0	0	0	0	0	0	0	0
16th Data	0	0	0	0	0	0	0	0

Figure 3.1.1 Device ID

FT : Function type of peripheral device
FD1 : First function's function definition block
FD2 : Second function's function definition block
FD3 : Third function's function definition block

FT_0-FT_{31} : Indicates function type of peripheral device. There are 32 types in total.

FD_{31}-FD_0 : Function definition blocks
These are blocks used for defining elements of the functions

4 Data format

The following describes the keyboard function data format. The table indicates the memory image at the host.

4.1 Write format

This is the format for writing data to keyboard function. When the host sends out write data using "Set Condition," the keyboard will return "Device Reply" for normal termination, or error code if error occurs.

bit	7	6	5	4	3	2	1	0
1st Data	LD_7	LD_6	LD_5	LD_4	LD_3	LD_2	LD_1	LD_0
2nd Data	0	0	0	0	0	0	0	0
3rd Data	0	0	0	0	0	0	0	0
4th Data	0	0	0	0	0	0	0	0

Figure 4.1.1 Write format

The write format is:

1st : LD LED setting
2nd : W1 Reserved = 00h
3rd : W2 Reserved = 00h
4th : W3 Reserved = 00h

LED setting

LD	Mapping	Light on	Light off
0	Num Lock	1	0
1	Caps Lock	1	0
2	Scroll Lock	1	0
3	Reserved	1	0
4	Reserved	1	0
5	Kana	1	0
6	Power	1	0
7	Shift	1	0

Figure 4.1.2 LED information bit

When light-on is set, the light will remain on until light-off is set. However, because the keyboard in this specification does not have any LEDs, **all are treated as light off (LD=`00h')**.

4.2 Read format

This is the data format for use when reading keyboard function data. When the host sends out "Get Condition," the keyboard will return "Data Transfer" data. The data size of the data format is 8 bytes.

bit	7	6	5	4	3	2	1	0
1st Data	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0
2nd Data	LD_7	LD_6	LD_5	LD_4	LD_3	LD_2	LD_1	LD_0
3rd Data	$KC1_7$	$KC1_6$	$KC1_5$	$KC1_4$	$KC1_3$	$KC1_2$	$KC1_1$	$KC1_0$
4th Data	$KC2_7$	$KC2_6$	$KC2_5$	$KC2_4$	$KC2_3$	$KC2_2$	$KC2_1$	$KC2_0$
5th Data	$KC3_7$	$KC3_6$	$KC3_5$	$KC3_4$	$KC3_3$	$KC3_2$	$KC3_1$	$KC3_0$
6th Data	$KC4_7$	$KC4_6$	$KC4_5$	$KC4_4$	$KC4_3$	$KC4_2$	$KC4_1$	$KC4_0$
7th Data	$KC5_7$	$KC5_6$	$KC5_5$	$KC5_4$	$KC5_3$	$KC5_2$	$KC5_1$	$KC5_0$
8th Data	$KC6_7$	$KC6_6$	$KC6_5$	$KC6_4$	$KC6_3$	$KC6_2$	$KC6_1$	$KC6_0$

Figure 4.2.1 Read format

The read format is:

1st : Modified key bit
2nd : LED information
3rd : Key code array #1
4th : Key code array #2
5th : Key code array #3
6th : Key code array #4
7th : Key code array #5
8th : Key code array #6

Modified key bit

M	Key Mapping	Down	Up
0	Left Ctrl	1	0
1	Left Shift	1	0
2	Left Alt	1	0
3	~~Left GUI~~S1	1	0
4	Right Ctrl	1	0
5	Right Shift	1	0
6	Right Alt	1	0
7	~~Right GUI~~S2	1	0

Figure 4.2.2 Modified key bit

Example:[Ctrl] + [Alt]

Normal '00000000' (00h)
Left Alt Down '00000100' (04h)
Right Ctrl Down '00010100' (14h)

LED information

LD	Mapping	Light on	Light off
0	Num Lock	1	0
1	Caps Lock	1	0
2	Scroll Lock	1	0
3	Reserved	1	0
4	Reserved	1	0
5	Kana	1	0
6	Power	1	0
7	Shift	1	0

Figure 4.2.3 LED information bit

Key code array

A key code is stored when a key on the keyboard is pressed. For a list of the key codes, see "5 Keycode." The key code array can store up to the 6th key pressed; if more keys are pressed (i.e., the 7th key), then the rollover error will be returned.

Key status	Modified key	Array#1	Array #2	Array #3	Array #4	Array #5	Array #6	Description
Normal	00h	00h	00h	00h	00h	00h	00h	
Right [Alt] Down	40h	00h	00h	00h	00h	00h	00h	Only modified key is updated.
[A] Down	40h	04h	00h	00h	00h	00h	00h	Array is packed from the front.
[H] Down	40h	04h	0Bh	00h	00h	00h	00h	
[I] Down	40h	04h	0Bh	0Ch	00h	00h	00h	
[M] Down	40h	04h	0Bh	0Ch	10h	00h	00h	
Left [Shift] Down	42h	04h	0Bh	0Ch	10h	00h	00h	Only modified key is updated.
[O] Down	42h	04h	0Bh	0Ch	10h	12h	00h	
[T] Down	42h	04h	0Bh	0Ch	10h	12h	17h	
[Q] Down	42h	01h	01h	01h	01h	01h	01h	Rollover error results because the 7th key is pressed.
[A] Up	42h	0Bh	0Ch	10h	12h	17h	00h	Array is packed from the front.
[H]+[M] Up	42h	0Ch	12h	17h	00h	00h	00h	
Right [Alt] Up	02h	0Ch	12h	17h	00h	00h	00h	Only modified key is updated.
[I] Up	02h	12h	17h	00h	00h	00h	00h	
[O] Up + [Y] Down	02h	17h	1Ch	00h	00h	00h	00h	Array is packed from the front, with Up having the higher priority.
[T] Up	02h	1Ch	00h	00h	00h	00h	00h	
Left [Shift] Up	00h	1Ch	00h	00h	00h	00h	00h	Only modified key is updated.
[Y] Up	00h	00h	00h	00h	00h	00h	00h	No key is pressed.

Figure 4.2.4 Example on operations

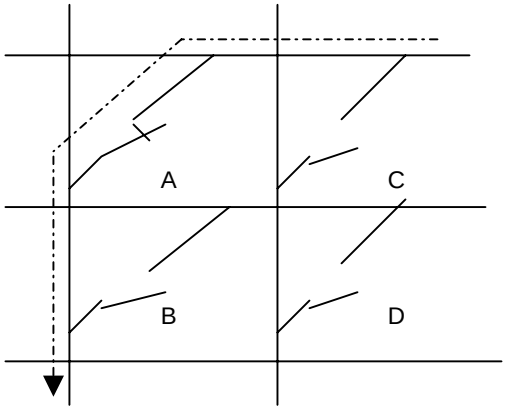
Boomerang effect

This problem happens when three or more keys are pressed simultaneously in a certain position. This confuses the keyboard circuitry so that key codes output are different than the actual keys pressed. When this occurs, the rollover error is returned.

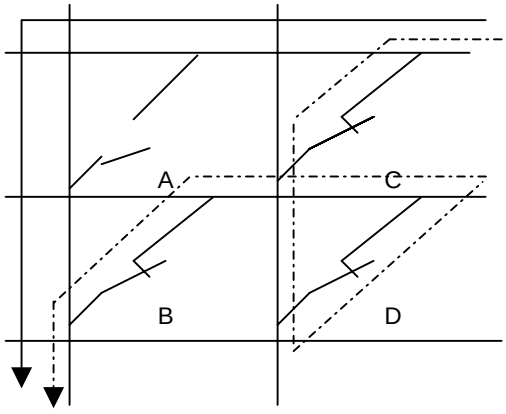
Key status	Modified key	Array#1	Array #2	Array #3	Array #4	Array #5	Array #6	Description
Normal	00h	00h	00h	00h	00h	00h	00h	
Right [Alt] Down	40h	00h	00h	00h	00h	00h	00h	Only modified key is updated.
[A] Down	40h	04h	00h	00h	00h	00h	00h	Array is packed from the front.
[Z] Down	40h	04h	1Dh	00h	00h	00h	00h	
[PS] Down	40h	~~01h~~ 04h	~~01h~~ 1Dh	01h	01h	01h	01h	Key causing the boomerang effect is present in key combination. Rollover error results.

Figure 4.2.5 Boomerang effect

The boomerang effect occurs in the way illustrated in figure 4.2.6.



A key is ON



B, C, D keys are ON

(Arrows of solid line and broken line have same value.)

Figure 4.2.6 Boomerang effect

In figure 4.2.6, boomerang effect occurs when current flows in the direction of the arrow as the three keys are pressed at the same time. From the standpoint of the detector, it looks like a key is pressed even though it is not.

5 Key code

Key code		Mapping/Usage
Decimal	Hexadecimal	
0	00h	No operation[6]
1	01h	Rollover error[6]
2	02h	POST Fail[6]
3	03h	Undefined error[6]
4	04h	Keyboard [a],[A][3]
5	05h	Keyboard [b],[B]
6	06h	Keyboard [c],[C][3]
7	07h	Keyboard [d],[D]
8	08h	Keyboard [e],[E]
9	09h	Keyboard [f],[F]
10	0Ah	Keyboard [g],[G]
11	0Bh	Keyboard [h],[H]
12	0Ch	Keyboard [I],[I]
13	0Dh	Keyboard [j],[J]
14	0Eh	Keyboard [k],]K]
15	0Fh	Keyboard [l],[L]
16	10h	Keyboard [m],[M][3]
17	11h	Keyboard [n],[N]
18	12h	Keyboard [o],[O][3]
19	13h	Keyboard [p],[P][3]
20	14h	Keyboard [q],[Q][3]
21	15h	Keyboard [r],[R]
22	16h	Keyboard [s],[S][3]
23	17h	Keyboard [t], [T]
24	18h	Keyboard [u],[U]
25	19h	Keyboard [v],[V]
26	1Ah	Keyboard [w],[W][3]
27	1Bh	Keyboard [x],[X][3]
28	1Ch	Keyboard [y],[Y][3]
29	1Dh	Keyboard [z], [Z][3]
30	1Eh	Keyboard [1], [!][3]
31	1Fh	Keyboard [2],[“][3]
32	20h	Keyboard [3],[#][3]
33	21h	Keyboard [4],[$][3]
34	22h	Keyboard [5],[%][3]
35	23h	Keyboard [6],[&][3]

Figure 5.1 Key code(1)

Key code		Mapping/Usage
Decimal	Hexadecimal	
36	24h	Keyboard [7],['][3]
37	25h	Keyboard [8],[(][3]
38	26h	Keyboard [9],[)][3]
39	27h	Keyboard [0],[~][3]
40	28h	Keyboard [Return](Enter)[4]
41	29h	Keyboard [Esc]
42	2Ah	Keyboard [Delete](Backspace)[10]
43	2Bh	Keyboard [Tab]
44	2Ch	Keyboard [Spacebar]
45	2Dh	Keyboard [-],[=][3]
46	2Eh	Keyboard [^],[¯][3]
47	2Fh	Keyboard [@],[`][3]
48	30h	Keyboard [[],[{][3]
49	31h	Not used for Japanese keyboard
50	32h	Keyboard []],[}][2]
51	33h	Keyboard [;],[+][3]
52	34h	Keyboard [:],[*][3]
53	35h	Keyboard Hankaku/Zenkaku
54	36h	Keyboard [,],[<][3]
55	37h	Keyboard [.],[>][3]
56	38h	Keyboard [/],[?][3]
57	39h	Keyboard [Caps Lock][8]
58	3Ah	Keyboard [F1]
59	3Bh	Keyboard [F2]
60	3Ch	Keyboard [F3]
61	3Dh	Keyboard [F4]
62	3Eh	Keyboard [F5]
63	3Fh	Keyboard [F6]
64	40h	Keyboard [F7]
65	41h	Keyboard [F8]
66	42h	Keyboard [F9]
67	43h	Keyboard [F10]
68	44h	Keyboard [F11]
69	45h	Keyboard [F12]
70	46h	Keyboard [Print Screen][1]
71	47h	Keyboard [Scroll Lock][8]
72	48h	Keyboard [Pause][1]
73	49h	Keyboard [Insert][1]
74	4Ah	Keyboard [Home][1]

Figure 5.2 Key code(2)

Key code		Mapping/Usage
Decimal	Hexadecimal	
75	4Bh	Keyboard [Page Up][1]
76	4Ch	Keyboard [Delete Forward][1,11]
77	4Dh	Keyboard [End][1]
78	4Eh	Keyboard [Page Down][1]
79	4Fh	Keyboard [→][1]
80	50h	Keyboard [←][1]
81	51h	Keyboard [↓][1]
82	52h	Keyboard [↑][1]
101	65h	Keyboard [~~Application~~S3][~~7~~]
135	87h	Keyboard Kanji [\],[_]
136	88h	Keyboard Kanji [Katakana],[Hiragana]
137	89h	Keyboard Kanji [~~¥~~],[|]
138	8Ah	Keyboard Kanji [Henkan]
139	8Bh	Keyboard Kanji [Muhenkan]
*224	E0h	Keyboard Left [Ctrl]
*225	E1h	Keyboard Left [Shift]
*226	E2h	Keyboard Left [Alt]
*227	E3h	Keyboard ~~Left~~ [~~GUI~~S1][~~13~~]
*228	E4h	Keyboard Right [Ctrl]
*229	E5h	Keyboard Right [Shift]
*230	E6h	Keyboard Right [Alt]
*231	E7h	Keyboard ~~Right~~ [~~GUI~~S2][~~13~~]

Figure 5.3　Key code(3)

*: The 2nd data of the read format for these keys is handled as a change key. Therefore, no key code is returned for the 3rd data to the 8th data.

Note:

1. Key code does not change even when the [Ctrl], [Alt], [Shift], or [Num Lock] is pressed..
   The status of the [Ctrl], [Alt], [Shift], or [Num Lock] key does not add any new key code.
2. Mapping differs depending on the keyboard language in use.
3. Mapping is performed for other language on the host.
4. The Enter key on the keyboard and the Enter key on the numeric keypad have different key codes.
5. Some are reserved for the particular keyboard language, e.g. FEP, IME.
6. Reserved for normal keyboard status or keyboard error. It's taken as a keyboard array, but physically no key is pressed.
7. ~~Windows application keys~~ Application keys
8. As a non-fixed key
9. As a fixed-key. It is used as a toggle button in order to support the older generation machines. Non-fixed key is recommended.
10. Move the cursor back by one position while deleting one character.
11. Delete the previous character without moving the cursor position.
12. Toggle between Zenkaku and Hankaku.
13. ~~Windows environment keys, e.g. left Windows key~~
14. ~~Windows environment keys, e.g. right Windows key~~

6 Keyboard function information

The following describes the device status. The device status cannot be modified or deleted, but it holds the data as it is.

6.1 Type

Fixed Device Status

This is a 112-byte, fixed format device status data that must be present. If all of the items are not recorded, then the operation and connection cannot be guaranteed.

Free Device Status

This is a status that can freely be used by individual devices, with a maximum capacity of 908~~912~~ bytes.

6.2 Fixed Device Status

Fixed Device Status must record all of the following items:

Device ID

Data size : 16 bytes (0000004001020099800000000000000000)

Description : This records the device ID of the peripheral.

Country of specification

Data size : 1 byte (02h)

Description : Japan only

Connection direction

Data size : 1 byte

Description : This records the direction of the expansion socket for connecting expansion device.
Because keyboard is not connected to the expansion device, the value is 00h.

Product Name

Data size : 30 bytes

Description : "92key Keyboard for JPN"
Use space code (20h) to pad remaining space.

License

Data size : 60 bytes

Description : "Produced By or Under License From SEGA ENTERPRISES,LTD."
Use space code (20h) to pad remaining space.

Standby current

Data size : 2 bytes

Description : Use hexadecimal to record the standby current in 0.1mA unit.
Because "92key Keyboard" is ~~30~~35mA, it should be recorded as 01-~~2Ch~~5Eh.

Maximum current

Data size : 2 bytes

Description : Use hexadecimal to record the maximum current in 0.1mA unit.

Because "92key Keyboard" is ~~30~~40mA, it should be recorded as 01-~~2C~~90h.

6.3 Free Device Status

Free Device Status is an area where product planners, developers, designers, or programmers can write data to. This status can be obtained by issuing the All Device Request command from the host. When using it in application, be sure it can handle data array, as follows:

"Version 1.000,1998/~~xx~~06/~~xx~~12, 315-~~xxxx~~6125-A~~B~~D ,"
"Key Scan Module: The 1st Edition. 05/20"
~~"92key Keyboard for JPN"~~
~~"M.K" Chip~~
~~"Produced By or Under License From SEGA ENTERPRISES,LTD."~~

as written in Hankaku.

7 Epilogue

Before the release version Rev. 1.0 is distributed, part or most of the content may be changed.