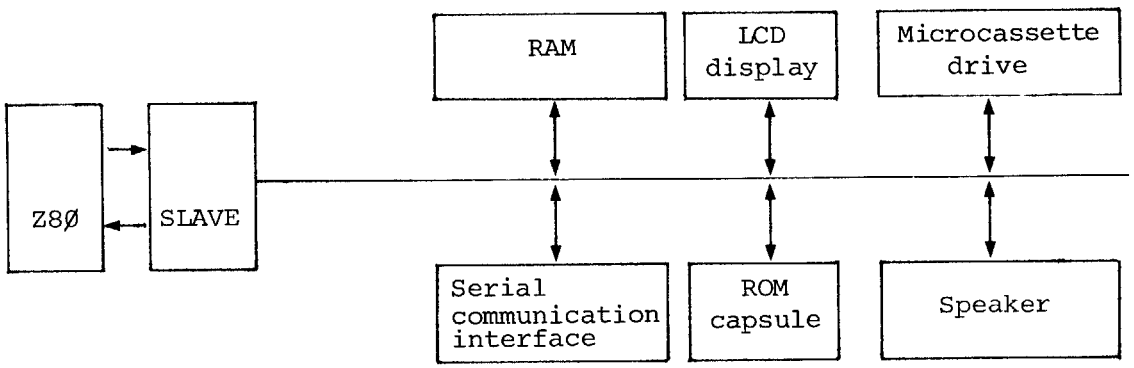
Chapter 13 6301 Slave CPU Operations

13.1 Functions

The 6301 slave CPU controls the following six types of devices:

- RAM
- LCD display
- Microcassette drive
- Serial communication interface
- ROM capsule
- Speaker

The slave CPU runs on its own control programs so that the Z80 CPU need only issue commands to the slave system to control the above devices. The Z80 commands to the slave CPU are detailed in Section 13.5.



(1) RAM

The slave CPU has 6K bytes of RAM which is located at slave system memory locations 8000H and higher. This RAM contains the VRAM and external character areas; it is mainly used for screen-oriented operations. The RAM is also loaded with the control programs for the slave CPU. The memory configuration of this RAM is shown on page 13-3.

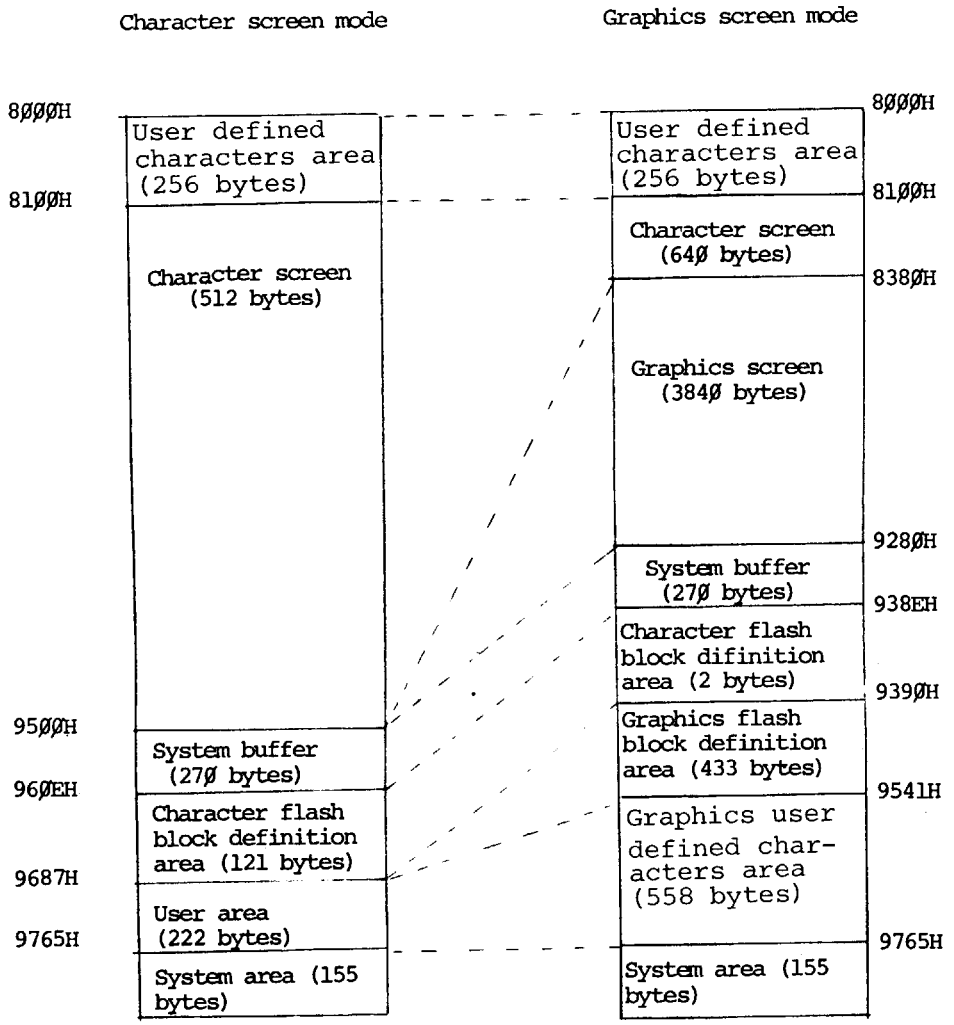
(2) LCD display

1) Theory of operation

The data to be displayed on the LCD is fetched by the LCD controller from RAM for display; the Z80 CPU need only place the display data into the specified RAM area.

There are two screen modes: the character and graphics modes. In the character mode, the LCD controller receives 1-byte character codes from the character screen area and searches the character generator in the controller for the corresponding (6 x 7 dots) font for display on the LCD. For user defined characters, it searches the user defined characters area at the beginning of the RAM for display.

Slave System Memory Map

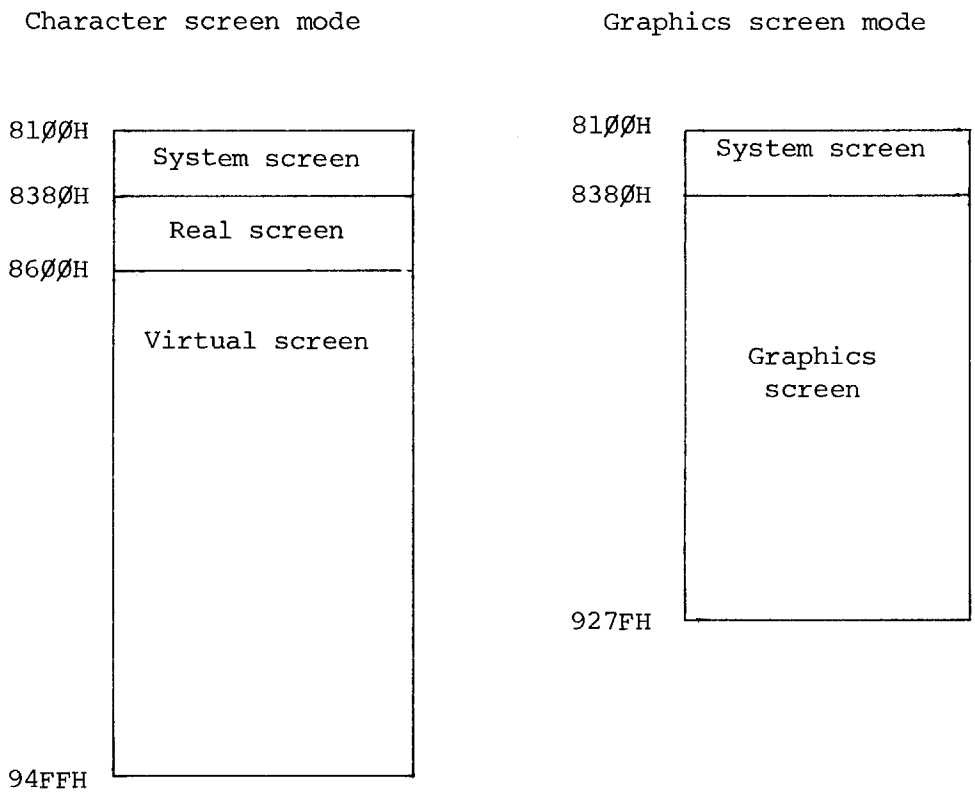


Japanese-language kanji characters are displayed in the graphics screen mode.

In the graphics mode, each single dot on the LCD is associated with a single bit in the graphics screen area in the RAM.

2) Screen configuration

The LCD screen configuration is shown below.



The system screen is used by the OS and not available to application programs. In the system screen, data is always displayed using character codes, independent of the screen mode. It has the same size as the LCD (80 x 8 = 640 bytes).

The real screen is as large as the 80 x 8 dot LCD and holds a portion of the virtual screen. Its image is displayed on the LCD by the LCD controller.

Since each dot on the LCD corresponds to a single data bit in the graphics screen and the LCD measures 480 dots by 64 dots, the LCD requires 480 dots x 64 dots ÷ 8 bits/byte = 3840 bytes of RAM. The graphics screen holds characters in bit image so that they may be displayed simultaneously with graphics data. See Chapter 6, "CONOUT" for further information.

3) Screen-related RAM areas

a) User defined characters area

The user can define characters in this area. It holds up to 32 user defined characters from 0E0H through 0FFH. User defined characters are defined using the ESC - 0E0H sequence via CONOUT.

b) Character screen area

The character screen area is used to store codes of characters to be displayed. In the screen configuration diagram in 2) above, all screen areas in the character screen mode and the system screen area in the graphics screen mode are used as the character screen area.

c) Graphics screen area

The graphics screen area is used to store data to be displayed in dot image. Its size is zero in the character screen mode.

The system buffer is used by the system when exchanging data with the MCT, serial communication interface, and Z80 CPU.

e) Character flash block definition area

The character flash block definition area contains character data to be flashed on the character screen. Three bytes are required to flash a single character (2-byte address and the character to be flashed) so a maximum of 40 characters can be flashed at a time in the character mode. Flashing is not available in the graphics screen mode (since the graphics screen holds character data in bit image form, it can use the graphics flash definition area that is described below). A character flash definition area can be defined by issuing the command code 31H to the slave CPU.

f) Graphics flash block definition area

The graphics flash block definition area contains graphics data to be flashed on the graphics screen. Three bytes are required to flash 8 consecutive dots

(2-byte address and 1-byte graphics data to be flashed) so a maximum of 144 dots can be flashed at a time in the graphics mode. This feature is available only in the graphics screen mode. Since the graphics screen holds graphics data in bit image form, it can use the graphics flash definition area that is described below. A graphics flash block definition area can be defined by issuing the command code 21H to the slave CPU.

g) Graphics user defined character definition area

The graphics user defined character definition area contains user defined character data for display on the graphics screen. This area can be used only in the graphics mode. A graphics user defined character definition area can be defined by issuing the command code 20H to the slave CPU.

h) User area

The user area (the remainder of the graphics user defined character definition area in the graphics screen mode) is not used by the slave CPU programs. The user may load programs into this area for execution.

i) System area

The system area is used by the slave CPU programs.

(3) Microcassette drive

The application program can perform various operations on the microcassette using commands described in Section 13.5. Since, however, files on the microcassette drive is controlled all by MTOS, an error would occur if an application program issued a microcassette command directly to the slave CPU while it was performing an I/O operation to the microcassette drove. No application program is therefore allowed to control the microcassette drive using the slave subsystem.

The application program may, however, exert direct control over the microcassette drive when playing back audio data tape (playback is not controlled by MTOS because no file operation is involved). The following precaution must be observed when driving a microcassette drive directly from an application program:

Precaution: Be sure to execute motor stop (4AH) and head off (42H) commands in this sequence before issuing an MCT-related command to the slave CPU.

To playback a microcassette tape, execute the following commands sequentially:

- Head On (command code 41H)
- Play (command code 48H)

(4) Serial communication interface

The MAPLE CP/M supports communication via a serial interface. Its slave CPU supports EPSP for controlling communication with external floppy disk drives.

(5) ROM capsule

The MAPLE can read data from the ROM capsules attached to the rear panel of the main unit. See Chapter 15 for the ROM memory map for the ROM capsules.

(6) Speaker

The MAPLE provides several functions to drive the speaker.

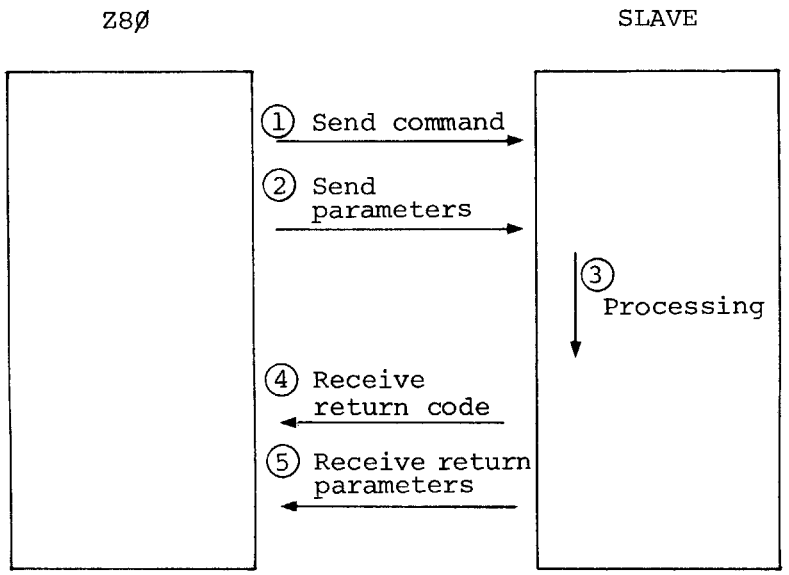
13.2 Data Backup

Since the MAPLE supports continue mode operations, the slave CPU must preserve the system status before power is turned off and restores it immediately after power is turned on again. To achieve this, the slave CPU provides the following functions:

- Battery backup of 6K RAM. The RAM data is always maintained whether MAPLE power is turned on or off.
- Saving and restoring the contents of CPU registers. These functions are used only by the OS POWER ON/OFF subroutines and not accessible to application programs (so they are not covered in this manual).

13.3 Z80-to-slave-CPU Communication Procedure

The Z80 and slave CPU communicate using the procedure described below.



The command (1) and return code (4) are 1 byte long and must always be issued. The presence and length of parameters (2 and 4) may differ depending on the command. The minimum parameter length is Ø, that is, no parameter. If the received return code contains a nonzero value (abnormal termination), the calling program will receive no return parameter.

Every sequence of steps 1 through 5 or 4 above must always be concluded; that is, no subsequent command must be issued before step 5 (or 4) is completed (the system will hang up if attempted).

The application program communicate with the slave CPU through the BIOS SLAVE function (WBOOT + 72H) (see Chapter 4, "BIOS Call" for details).

A command packet for turning on the speaker is shown below.

```
SPON:
        LD      A,0FFH
        LD      (SLVFLG),A
        LD      DE,072H
        LD      HL,(1)
        ADD     HL,DE
        LD      DE,PACK1  <-------- Communication packet address
        JP      (HL)      <-------- BIOS call (Call SLAVE.)
```

```
Communication
packet
PACKE1: DW      SENDSV    <-------- Send packet address
        DW      SENDLN    <-------- Send packet length
        DW      RCVSV     <-------- Receive packet address
        DW      RCVLNG    <-------- Receive packet length
SENDSV: DE      72H,80H   <-------- Send packet (command + send
                                     parameters)
SENDLN: EQU     2         <-------- Send command + parameters
RCVSV:  DS      1         <-------- Receive packet
RCVLNG: EQU     1         <-------- Receive packet length
                                     (return code only)
```

Note: Two-byte address parameters must be sent to the slave CPU in higher-order-byte-first-sequence. This is because the slave CPU (6301) handle the 2-byte data the higher-order-byte-first as opposed to the Z80 CPU.

13.4 Slave CPU Commands

The rest of this chapter describes the slave CPU commands arranged by device.

How to interpret the command table

- The send packet length is the length of the command (one byte) plus its send parameters (SP1, SP2, ... SPn in 1)).
- The receive packet length is the length of the return code (one byte) plus receive parameters (RP1, RP2, ... RPn in 2)).
- The return code in 3) identifies the type of the status code returned by the slave CPU. The return code table is given at the end of this chapter.
- Commands are represented in hexadecimal notation. All parameters are treated as binary data.
- "Physical screen" refers to the "real screen" used in previous sections.

(1) RAM

This subsection explains the monitor program which controls program execution and RAM access.

Table 3-1 RAM commands

Code	Function
00	Read data
01	Write data
02	Execute routine

(1-1) Read Data (ØØ)

1) Send parameter

SP1: Address (high)

SP2: Address (low)

2) Receive parameter

RP1: Read data

3) Return code

RCDØØ

4) Function

Reads the data at the location designated by SP1 and SP2.

5) Note

An attempt to read data at a location lower than 8ØH will destroy the system.

(1-2) Write Data (Ø1)

1) Send parameter

SP1: Address (high)

SP2: Address (low)

SP3: Write data

SP4: Operation {1: AND, 2: OR, 3: XOR

Others: Store only

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Performs the specified operation on the data and places the results at the address designated by SP1 and SP2.

(1-3) Execute Routine (Ø2)

1) Send parameter

SP1: Address (high)

SP2: Address (low)

2) Receive parameter

None.

3) Return code

?

4) Function

Transfers control to the address designated by SP1 and SP2. The following registers are loaded with as follows:

IX (97F4, 97F5)

A (97F3)

B (97F2)

C (97F1)

(2) Screen

This subsection describes the screen-related commands used to control the graphics and character screens.

Table 3-2 Screen Commands

Code	Function
10	Defines screen mode.
11	Turns on/off LCD.
12	Selects screen.
13	Reads screen pointer.
14	Sets screen pointer.
15	Defines number of lines.
16	Defines cursor mode.
17	Reads cursor position.
18	Defines cursor position.
19	Starts/Stops control block flashing.
1A	Clears screen.
1B	Reads character font.
20	Defines user graphics character.
21	Defines graphics screen block flashing data.
22	Draws character font on graphics screen.

23	Draws user defined graphics character on graphics screen.
24	Reads graphics screen data.
25	Displays data on graphics screen.
26	Moves graphics screen block.
27	Sets point.
28	Reads point.
29	Draws line.
30	Defines user character.
31	Defines character screen block flashing data.
32	Reads window pointer.
33	Sets window pointer.
34	Reads character screen data.
35	Displays data on character screen.
36	Moves character screen block.

(2-1) Define screen mode (10)

1) Send parameter

SP1 SP2: Character screen starting address

SP3 SP4: Graphics screen starting address

SP5 SP6: Character flash block starting address

SP7 SP8: Graphics flash block starting address

SP9 SP10: Graphics user defined character buffer starting address

SP11: Number of lines on character screen

SP12: User defined character starting code

SP13 SP14: Communication buffer address

SP15: LCD status

SP16: Screen status

2) Receive parameter

None.

3) Return code

RCD00

4) Function

Defines the configuration of the VRAM. The user cannot use this command since the VRAM configuration is defined by the OS as described on page 13-3. The OS display routine (CONOUT) will not function normally if the VRAM configuration is defined by the user.

(2-2) Turn On/Off LCD (11)

1) Send parameter

SP1: ON/OFF switch (ØØ: OFF, Others: ON)

2) Receive parameter

None.

3) Return code

RCD00

4) Function

Turns on or off the LCD display. This command only specifies whether data is to be displayed on the LCD and do not affect the contents of VRAM at all.

(2-3) Select Screen (12)

1) Send parameter

SP1: Screen select code (ØØ: Graphics screen,
Others: Character screen)

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Selects the screen to be displayed on the physical screen. The screen is set to the window pointer position when the character screen is selected and set to the top when the graphics screen is selected.

(2-4) Read Screen Pointer (13)

1) Send parameter

None.

2) Receive parameter

RP1: Address (high)

RP2: Address (low)

3) Return code

RCD00

4) Function

Reads the starting address of the VRAM area whose contents are currently displayed on the physical screen.

5) Note

The LCD controller accesses VRAM only between addresses 8000H and 97FFH. Therefore, the highest order three bits of the pointer is insignificant and always set to 100.

(2-5) Set Screen Pointer (14)

1) Send parameter

SP1: Address (high)

SP2: Address (low)

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Sets the physical screen starting address designated by SP1 and SP2.

The valid addresses are 8000H to 97FFH.

The LCD controller displays 640 bytes of data starting at the address pointed to by the pointer in the character screen mode. It displays 3840 bytes of data in the graphics mode.

(2-6) Define Number of Lines (15)

1) Send parameter

SP1: ØØ: 8 lines

Others: 7 lines

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Defines the number of lines to be displayed in the character screen mode. If this command is issued when the graphics screen mode is selected, the value defined by the command becomes valid when the screen mode is switched to character screen.

8 or 7 lines refer to the number of lines to be displayed on the LCD. The OS always uses 8-line-per-screen mode. Characters are displayed with a wider spacing in 7-line mode than in 8-line mode.

(2-7) Define Cursor Mode (16)

1) Send parameter

SP1: Bit 7 - Bit 3: Ø

Bit 2: Cursor font (Ø: Under line, 1: Block)

Bit 1: Cursor blink (Ø: OFF, 1: ON)

Bit Ø: Cursor ON/OFF (Ø: OFF, 1: ON)

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Defines the cursor mode to be used in the character screen mode according to the settings of bits Ø to 2 of SP1.

This command is valid only in the character screen mode. In the graphics mode, the cursor is always displayed in the underlined blink mode.

(2-8) Read Cursor Position (17)

1) Send parameter

None.

2) Receive parameter

RP1: Cursor X (Ø - 79)

RP2: Cursor Y {7-line mode (Ø - 6)

8-line mode (Ø - 7)

3) Return code

RCDØØ

4) Function

Reads the cursor position on the physical screen (in the character screen mode).

The upper left corner of the screen is taken as coordinates (Ø,Ø).

(2-9) Set Cursor Position (18)

1) Send parameter

SP1: Cursor X (Ø - 79)

SP2: Cursor Y {7-line mode (Ø - 6)

8-line mode (Ø - 7)

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Sets the cursor position on the physical screen (in the character screen mode).

The cursor position must be within the maximum line and column numbers of the physical screen.

(2-10) Start/Stop Control Block Flashing (19)

1) Send parameter

SP1: 00: Stop

Others: Start

(01, ... FF)

The unit is 100 msec.

2) Receive parameter

None.

3) Return code

RCD00

4) Function

Turns on and off block flashing on the current screen.

If no flash block is specified, this command does nothing even if block flashing is turned on.

The block to be flashed changes as the screen mode is changed.

Block flashing is specified by issuing a command 21H (in the graphics screen mode) or 31H (in the character screen mode).

(2-11) Clear Screen (1A)

1) Send parameter

SP1: Screen to be cleared

ØØ: Graphics screen

Others: Character screen

SP2: Clear code

SP3: Starting line

SP4: Number of lines to be cleared.

2) Receive parameter

None.

3) Return code

RCDØØ

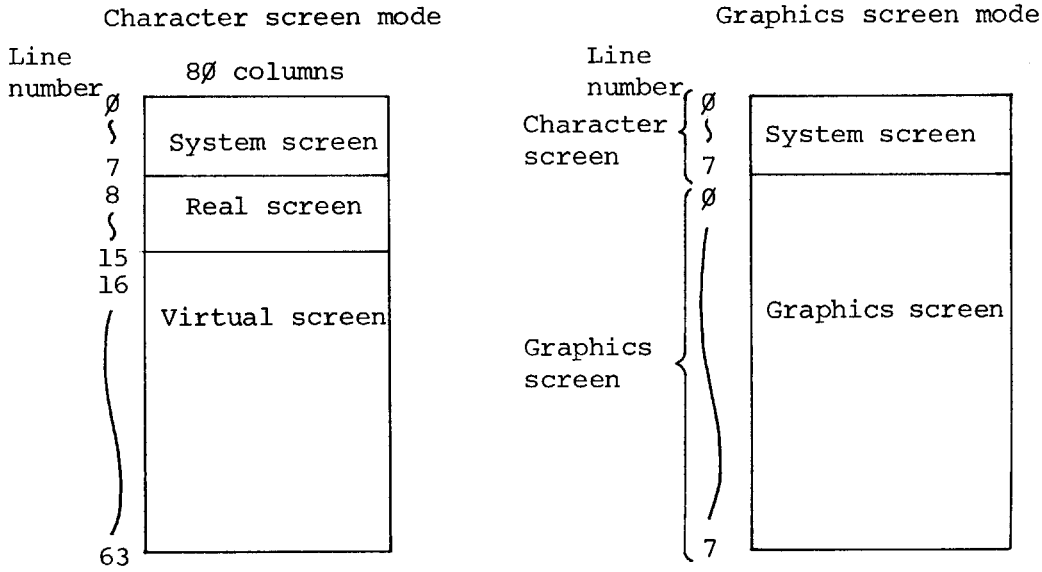
4) Function

Fills the specified number of lines (specified by SP4) starting at the line designated by SP3 on the screen area specified by PS1 with the code specified by SP2.

This command is invalid when the specified screen does not exist.

SP4 must not be set to 0. (Specifying 0 as SP4 value will destroy the system.)

The screen specified in SP1 refers to the screen shown in the memory map on page 13-3. The relationship between the line number and the screen type is shown below.


Character screen mode
Line number
8Ø columns
Ø
7
8
15
16
63
System screen
Real screen
Virtual screen
Graphics screen mode
Line number
Character screen
Graphics screen
Ø
7
Ø
7
System screen
Graphics screen

(2-12) Read Character Font (1B)

1) Send parameter

SP1: Character code

2) Receive parameter

RP1: Font data 1

RP2: Font data 2

RP3: Font data 3

RP4: Font data 4

RP5: Font data 5

RP6: Font data 6

RP7: Font data 7

RP8: Font data 8

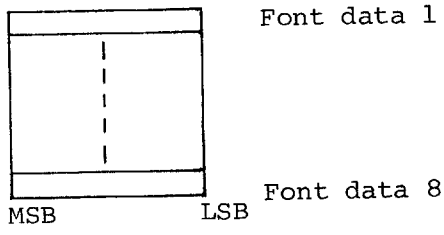
3) Return code

RCD00

4) Function

Reads the font of the specified character from the LCD controller or user defined character area.

5) Font image



Note: The correspondence between the character codes and fonts is shown in Chapter 20.

(2-13) Define User Defined Graphic Character (20)

1) Send parameter

SP1: Definition mode

00: Clear user defined character area

Others: User defined character code

SP2: Hight (size along x-axis)

SP3: Width (size along y-axis)

SP4: Data 1

.

.

SPn: Data n-3

2) Receive parameter

None.

3) Return codes

RCD00

RCD04

4) Function

Defines a graphics user defined character or clears the user defined character area. The definition code must not be 00.

Even if a user defined character is defined with a code which has already been assigned to another character, the character to be displayed on the graphics screen is the one

originally defined with that code (though the later definition is accepted.)

The sizes along x- and y-axes must be within the range x*y<=255. Specifying a size larger than that will cause an error. The definition will also be invalidated if too small x and y values are specified.

The user defined character block must be specified in byte units.

Data is placed in the area in row-first order.

Example:

x:3, y:4, Byte count: 12

Data 1	Data 2	Data 3
Data 4	Data 5	Data 6
Data 7	Data 8	Data 9
Data 10	Data 11	Data 12

(2-14) Define Graphics Screen Block Flashing Data (21)

1) Send parameter

SP1: Number of blocks

SP2: x-coordinate of the first block

SP3: y-coordinate of the first block

SP4: First block blink data

•

•

SPn-2: x-coordinate of the kth block

SPn-1: y-coordinate of the kth block

SPn: kth block blink data

2) Receive parameter

None.

3) Return codes

RCD00

RCD04

4) Function

Defines the coordinates of the blocks to flash and the blink data on the graphics screen. The block flashing data must not exceed the specified block. The maximum number of blocks that can be specified is 144.

(2-15) Draw Character Font on Graphics Screen (22)

1) Send parameter

SP1: x-coordinate (high)

SP2: x-coordinate (low) (Ø - 479)

SP3: y-coordinate (Ø - 63)

SP4: Character code (Ø - FF)

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Reads the font corresponding to a character code specified in SP4 from the character generator and draws the font on the screen using the coordinates designated by SP1, SP2, and SP3 as the origin.

Graphics coordinates must be specified in bits. The portion of the font extending beyond the right edge of the screen is entered into the screen from the left edge.

(2-16) Draw User Defined Character on Graphics Screen (23)

1) Send parameter

SP1: x-coordinate (0 - 59)

SP2: y-coordinate (0 - 63)

SP3: Graphics user defined character

2) Receive parameter

None.

3) Return codes

RCD00

RCD05

4) Function

Draws the block of the user defined character code specified in SP3 starting at the graphics screen coordinates specified in SP1 and SP2. Graphics user defined character can be defined by issuing a command 20H.

(2-17) Read Graphics Screen Data (24)

1) Send parameter

SP1: x-coordinate (Ø - 59)

SP2: y-coordinate (Ø - 63)

SP3: Byte count (Ø: 256, 1: 1, ... FF: 255)

2) Receive parameter

SP1: Data 1

.

.

SPn: Data n

3) Return codes

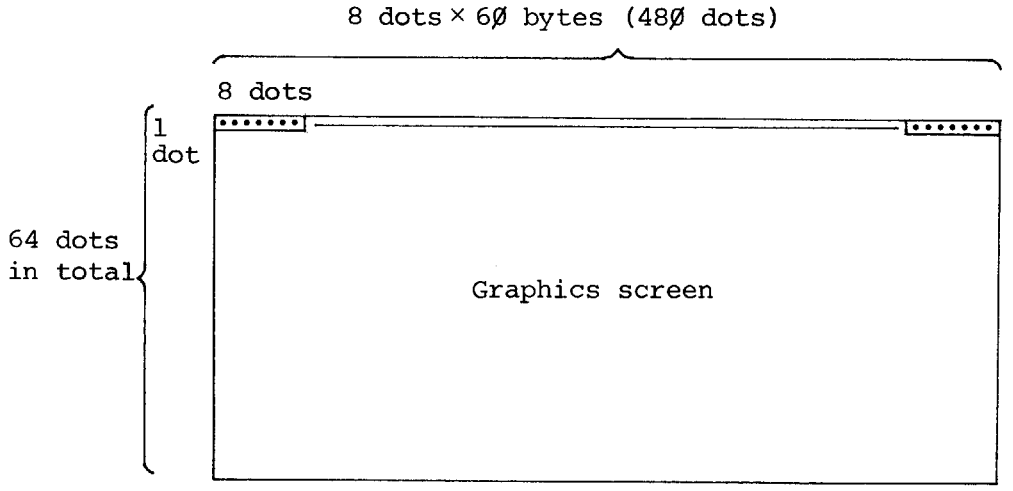
RCDØØ

RCDØ4

4) Function

Reads the number of bytes specified in SP3 in row-first order starting at the coordinates on the graphics screen designated by SP1 and SP2. Garbage data is returned for the portion of data which is specified in SP3 and which is out of the screen.

The number of receive parameters (n) is the same as that of data count specified in the third send parameter.


8 dots × 6Ø bytes (48Ø dots)
8 dots
1
dot
64 dots
in total
Graphics screen

(2-18) Display Data on Graphics Screen (25)

1) Send parameter

SP1: x-coordinate (Ø - 59)

SP2: y-coordinate (Ø - 63)

SP3: Hight (size along x-axis)

SP4: Width (size along y-axis) } x*y<=255

SP5: Operation ØØ: Store Ø1: AND

Ø2: OR Ø3: EOR

SP6: Data 1

•

•

SPn: Data n-5

2) Receive parameter

None.

3) Return codes

RCDØØ

RCDØ4

4) Function

Stores the block of data specified in SP3 and SP4 at the specified coordinates on the graphics screen. Data is sent in row-first order.

Data 1 Data 2

Data 3 Data 4

Data 5 Data 6 Size: x = 2, y = 3

This command performs the specified operation on the write data and writes the result as the new data.

When store operation is specified, the command writes data as is, without masking with a mask pattern. Drawing is stopped when the write data overflows the screen.

Both height (x) and width (y) of the block must be 1 or larger. The product of x and y must be 255 or less.

(2-19) Move Graphics Screen Block (26)

1) Send parameter

SP1: Source x-coordinate (Ø - 59)

SP2: Source y-coordinate (Ø - 63)

SP3: Size along X-axis

SP4: Size along Y-axis

SP5: Destination x-coordinate (Ø - 59)

SP6: Destination y-coordinate (Ø - 63)

2) Receive parameter

None.

3) Return codes

RCDØØ

RCDØ4

4) Function

Moves the block designated by SP1, SP2, SP3, and SP4 on the graphics screen starting at the coordinates specified by SP5 and SP6.

(2-20) Define Point (27)

1) Send parameter

SP1: x-coordinate (high) }
SP2: x-coordinate (low) } (Ø - 479)

SP3: y-coordinate (Ø -63)

SP4: Operation

Ø1: OFF Ø2: ON Ø3: Complement

2) Receive parameter

None.

3) Return codes

RCDØØ

RCDØ4

4) Function

Performs the specified operation on the dot at the specified graphic screen coordinates.

(2-21) Read Point (28)

1) Send parameter

SP1: x-coordinate (high)
SP2: x-coordinate (low) } (Ø - 479)

SP3: y-coordinate (Ø - 63)

2) Receive parameter

RP1: ØØ: off Others: ON

3) Return codes

RCDØØ

RCDØ4

4) Function

Reads the state of the dot at the specified graphics screen coordinates.

(2-22) Draw Line (29)

1) Send parameter

SP1: Starting x-coordinate (high)
SP2: Starting x-coordinate (low) } (0 - 479)

SP3: Starting y-coordinate (high)
SP4: Starting y-coordinate (low) } (0 - 63)

SP5: Ending x-coordinate (high)
SP6: Ending x-coordinate (low) } (0 - 479)

SP7: Ending y-coordinate (high)
SP8: Ending y-coordinate (low) } (0 - 63)

SP9: Operation vector (high)
SP10: Operation vector (low) } 0: No operation 1: Operation

SP1: Point mode 01: OFF 02: ON 03: Complement

2) Receive parameter

None.

3) Return code

RCD00

4) Function

Draws a line on the graphics screen between the starting and ending coordinates. The points are displayed in the point mode only when the MSB of the operation vector is set to 1. The operation vector is rotated to the left at setting every single dots.

The top left corner of the screen is set to (0, 0). The horizontal and vertical axes are defined as x-axis and y-axis, respectively. Data at the points outside the screen is not displayed.

(2-23) User Defined Character (30)

1) Send parameter

SP1: Character code (E0H-FFH)

SP2: Data 1

.

.

SP9: Data 8

2) Receive parameter

None.

3) Return codes

RCD00

RCD06

4) Function

Defines an user defined character in the user defined character area. Invalid codes not in the range 0E0H to 0FFH are ignored.

"A", for example, is defined with the following data:

	bit 7	6	5	4	3	2	1	bit Ø
Data 1	Ø	Ø	Ø	Ø	1	1	Ø	Ø
Data 2	Ø	Ø	Ø	1	Ø	Ø	1	Ø
Data 3	Ø	Ø	1	Ø	Ø	Ø	Ø	1
Data 4	Ø	Ø	1	Ø	Ø	Ø	Ø	1
Data 5	Ø	Ø	1	1	1	1	1	1
Data 6	Ø	Ø	1	Ø	Ø	Ø	Ø	1
Data 7	Ø	Ø	1	Ø	Ø	Ø	Ø	1
Data 8	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

(2-24) Define Character Screen Block Flashing Data (31)

1) Send parameter

SP1: Number of blocks

SP2: x-coordinate of the first block (0 - 79)

SP3: y-coordinate of the first block (0 - 63)

SP4: First block blinking data

.

.

SPn-2: x-coordinate of the kth block

SPn-1: y-coordinate of the kth block

SPn: kth block blinking data

2) Receive parameter

None.

3) Return codes

RCD00

RCD04

4) Function

Specifies the coordinates of the block to flash and the blink data on the character screen. Specifying coordinates outside the screen will make all definitions in this command invalid. Up to 40 blocks can be defined. All blocks will be cleared when the screen mode is altered.

(2-25) Read window pointer (32)

1) Send parameter

None.

2) Receive parameter

RP1: x-coordinate (Ø - 79)

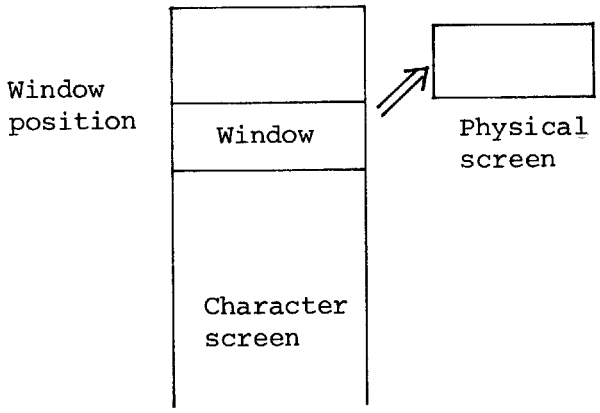
RP2: y-coordinate (Ø - 63)

3) Return code

RCDØØ

4) Function

Reads the character screen coordinates at which display is to begin.



(2-26) Set Window Pointer (33)

1) Send parameter

SP1: x-coordinate (Ø - 79)

SP2: y-coordinate (Ø - 63)

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Defines the window position on the character screen. This command is valid if issued in the graphics screen mode (it will be executed when the screen is switched to character mode).

(2-27) Read Character Screen Data (34)

1) Send parameter

SP1: x-coordinate (Ø - 79)

SP2: y-coordinate (Ø - 63)

SP3: Byte count (0:256, 1:1, ... FF:255)

2) Receive parameter

RP1: Data 1

•

•

RPn: Data n

3) Return code

RCDØØ

4) Function

Reads the specified number of data bytes from the character screen in row-first order. If the parameters are specified to read data beyond the character screen area, data other than display data is read.
The number of receive parameters (n) is the same as the number specified in the third send parameter.

(2-28) Display Data on Character Screen (35)

1) Send parameter

SP1: Starting x-coordinate (Ø - 79)

SP2: Starting y-coordinate (Ø - 63)

SP3: Byte count (1: 1, ... FF: 255)

SP4: Data 1

•

•

SPn: Data n-3

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Displays the specified data in row-first order on the character screen starting at the specified coordinates. The portion of data not fitting in the screen is ignored.

Byte count must not be set to Ø.

The relationship between the y-coordinate values and screen types is as follows:

Ø - 7: System screen

8 - 15: Real screen

16 - 63: Virtual screen

(2-29) Move Character Screen Block (36)

1) Send parameter

SP1: Source x-coordinate (Ø - 79)

SP2: Source y-coordinate (Ø - 63)

SP3: Hight (size along x-axis)

SP4: Width (size along y-axis)

SP5: Destination x-coordinate (Ø - 79)

SP6: Destination y-coordinate (Ø - 63)

2) Receive parameter

None.

3) Return codes

RCDØØ

RCDØ4

4) Function

Moves the block specified by SP1, SP2, SP3, and SP4 on the character screen from the coordinates designated by SP5 and SP6.

(3) Microcassette

This subsection describes the commands for driving the microcassette drive.

Table 3-3 Microcassette commands

Code	Command
40	Read microcassette status
41	Head on
42	Head off
43	Rewind n counts
44	Fast forward n counts
45	Rewind
46	Fast forward
47	Slow rewind
48	Play
49	Record
4A	Stop
4B	Read write protect pin
4C	Read counter
4D	Set counter
51	Write data and non-stop
52	Write data and stop
53	Read data and non-stop

54	Read data and stop
55	Set write protect area
56	pointer. Reset write protect area pointer

(3-1) Read Microcassette Status (40)

1) Send parameter

None.

2) Receive parameter

RP1: Microcassette status

Bit 7: Head position (0: OFF 1: ON)

Bit 6: Motor (0: Stop 1: Move)

Bit 5: Rewind (0: No 1: Wind)

Bit 4: FF (0: No 1: FF)

Bit 3: Play (0: No 1: Play)

Bit 2: Record (0: No 1: Record)

Bit 1:

Bit 0: Write protect area set flag

(0: Reset 1: Set)

3) Return code

RCD00

4) Function

Reads the status of a microcassette drive.

The write protect area setting flag (bit 0) is set to 1 when a command 55H is executed and set to 0 (reset) when a command 56H is executed.

(3-2) Head On (41)

1) Send parameter

None.

2) Receive parameter

None.

3) Return codes

RCD00

RCD07

4) Function

Turns the read/write head on.

This command must be issued while the microcassette drive is not in motion.

(3-3) Head Off (42)

1) Send parameter

None.

2) Receive parameter

None.

3) Return codes

RCDØØ

RCDØ7

4) Function

Turns the read/write head off.

This command must be issued while the microcassette drive is not in motion.

(3-4) Rewind n counts (43)

1) Send parameter

SP1: Counter value (high-order)

SP2: Counter value (low-order)

2) Receive parameter

None.

3) Return codes

RCD00

RCD07

RCD08

4) Function

Moves tape backward the number of tape count specified in SP1 and SP2.

5) Note

If tape is rewound 100 tape counts from count 1000, for example, the tape at position 900 will come under the read/write head.

The head is automatically turned off in the rewind mode.

(3-5) Fast Foward n Counts (44)

1) Send parameter

SP1: Counter value (high-order)

SP2: Counter value (low-order)

2) Receive parameter

None.

3) Return codes

RCD00

RCD07

RCD08

4) Function

Winds the tape forward by the counts specified by SP1 and SP2.

(3-6) Rewind (45)

1) Send parameter

None.

2) Receive parameter

None.

3) Return codes

RCDØØ

RCDØ7

4) Function

Rotates the motor in the reverse direction (rewind).

(3-7) Fast Foward (46)

1) Send parameter

None.

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Rotates the motor in the forward direction.

(3-8) Slow Rewind (47)

1) Send parameter

None.

2) Receive parameter

None.

3) Return codes

RCD00

RCD07

4) Function

Rotates the motor in the reverse direction at a low speed.

(3-9) Play (48)

1) Send parameter

None.

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Rotates the motor in the play mode.

The read signal is placed on the read line if the microcassette drive is in the head on state.

(3-10) Record (49)

1) Send parameter

None.

2) Receive parameter

None.

3) Return code

RCD00

4) Function

Rotates the motor in the record mode. Data on the tape will be erased if the read/write head is on.

(3-11) Stop (4A)

1) Send parameter

None.

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Stops the motor to rotate. Head position does not move.

(3-12) Read Write Protect Pin (4B)

1) Send parameter

None.

2) Receive parameter

RP1: ØØ: Protected

Others: Not protected

3) Return code

RCDØØ

4) Function

Used to check whether the current microcassette tape is write protected.

(3-13) Read Counter (4C)

1) Send parameter

None.

2) Receive parameter

RP1: Counter value (high-order)

RP2: Counter value (low-order)

3) Return code

RCD00

4) Function

Reads the current value of the 16-bit counter.

(3-14) Set Counter (4D)

1) Send parameter

SP1: Counter value (high-order)

SP2: Counter value (low-order)

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Sets the counter.

(3-16) Write Data and Non-stop (51)

1) Send parameter

SP1: Data length (high-order) } 0000:65536,

SP2: Data length (low-order) } 0001:1, ...

SP3: Pointer to block ID number

SP4: Data 1

.

.

SPn: Data n-3

2) Receive parameter

RP1: Block end counter (high-order)

RP2: Block end counter (low-order)

3) Return codes

RCD00

RCD07

RCD08

RCD09

4) Function

Writes the specified number of data bytes onto tape. A 00H in SP3 identifies the first block that is written for the current block and a 01H the second write for the block. This byte is automatically incremented by the slave CPU during processing. The issuing program must set this byte in the send data to 0FFH.

(3-17) Write Data and Stop (52)

1) Send parameter

SP1: Data length (high-order)

SP2: Data length (low-order)

SP3: Pointer to block ID number

SP4: Data 1

•

•

SPn: Data n-3

2) Receive parameter

RP1: Block end counter (high-order)

RP2: Block end counter (low-order)

3) Return codes

RCDØØ

RCDØ7

RCDØ8

RCDØ9

4) Function

This command performs the same function as Write Data and Non-stop except that it stops the tape after writing data.

(3-18) Read Data and Non-stop (53)

1) Send parameter

SP1: Data length (high-order)

SP2: Data length (low-order)

SP3: Block ID code

2) Receive parameter

RP1: Block start counter (high-order)*

RP2: Block start counter (low-order)

RP3: Data 1

•

•

RPn: Data n-2

3) Return codes

RCD00

RCD07

RCD08

RCD10

RCD11

RCD11-1

4) Function

Reads the specified bytes of data from tape. If an error is found in a block, this command returns a return codes RP1 and RP2, and starts reading the next block. If the specified block

is read normally, the command terminates reading of the block without stopping the motor.

The block ID code is a control code which is used to identify the beginning of a block.

No check is made on the block ID if 3FH ('?') is specified in SP3.

*: Counter value after preamble is read.

(3-19) Read Data and Stop (54)

1) Send parameter

SP1: Data length (high-order)

SP2: Data length (low-order)

SP3: Block ID code

2) Receive parameter

RP1: Block start counter (high-order)*

RP2: Block start counter (low-order)

RP3: Data 1

•

•

RPn: Data n

3) Return codes

RCD00

RCD07

RCD08

RCD10

RCD11

RCD11-1

4) Function

Reads the specified bytes of data from tape. This command has the same functions as Read Data and Non-stop except that it stops the motor after reading the specified block.

(3-20) Set Write Protect Area Pointer (55)

1) Send parameter

SP1: Counter value (high-order) } Protect area pointer
SP2: Counter value (low-order) }

2) Receive parameter

None.

3) Return code

RCD00

4) Function

Decrements the write protect pointer by 1 every time the tape counter increments by 1 at each read or write operation and terminates either read or write when the pointer reaches 0.
This command is used to stop processing on a cassette at a desired count. MTOS uses this function to erase a length of tape specified in the tape count parameter.
Whether processing has been stopped or not can be identified by examining bit 6 of the status information returned by command 40H. If this bit is 0, the motor has been stopped (that means processing has been terminated).
Whether the pointer is set or not can also be

identified by checking bit Ø of the status information returned by command 4ØH.

(3-21) Reset Write Protect Area Pointer (56)

1) Send parameter

None.

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Makes the value of the write protect area pointer invalid. After this command, the user can write on the tape without considering the write protect area pointer value.

(4) Serial I/O and serial communication

This subsection describes the commands for serial I/O and serial communication. Refer to Chapter 15 for the serial communications protocol.

Table 3-4 Serial I/O, serial communications commands

Code	Command
60	Read serial I/O port status
61	Set serial port bit rate
62	Serial input
63	Serial output
64	Send with header
65	Receive data

(4-1) Read Serial I/O Status (60)

1) Send parameter

None.

2) Receive parameter

RP1: Serial I/O status

Bit 7: Receive register status

(0: Empty 1: Full)

Bit 6: Overrun framing error

(0: No 1: Error)

Bit 5: Transmit register status

(0: Full 1: Empty)

Bit 4: Control out (0: Low 1: High)

Bit 3: Control in (0: Low 1: High)

3) Return code

RCD00

4) Function

Reads the serial I/O status value.

(4-2) Set Serial Port Bit Rate (61)

1) Send parameter

SP1: Specifies the bit rate, general purpose input port check bit, and general purpose output port level.

Bits 0 and 1: Sets bit rate.

	Bit 1	Bit Ø
38400	Ø	Ø
4800	Ø	1
600	1	Ø
150	1	1

Bit 5: General purpose output level.

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Sets the serial port bit rate and defines the level of the general purpose output port.

(4-3) Serial Input (62)

1) Send parameter

None.

2) Receive parameter

RP1: Data

3) Return codes

RCD00

RCD13

4) Function

Reads one character from the serial port.

(4-4) Serial Output (63)

1) Send parameter

SP1: Send data

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Sends one character to the serial port.

(4-5) Send Data with Header (64)

1) Send parameter

SP1: Receive data flag

(ØØ: Receive no data after sending.

Ø1: Receive data after sending.)

SP2: FMT

SP3: DID

SP4: SID = 22H

SP5: FNC

SP6: SI2

SP7: Data 1

•

•

SPn+7 : Data n

2) Receive parameter

(Valid only when SP1≧1.)

RP1: Header information (00: Header / 01: No header)

When SP1=00

FMT

DID

SID

FNC

SIZ

RP2: Data 1

.

.

RPk: Data n

3) Return codes

RCD00

RCD12

RCD13

RCD14

4) Function

Sends data with a header to the serial port according to the EPSP protocol. The command also receives data with a header if the receive data flag is set to 1. It terminates processing immediately when an error is detected.

(4-6) Receive Data with Header (65)

1) Send parameter

None.

2) Receive parameter

RP1: Header information (00: Header / 01: No header)

RP2: FMT
RP3: DID
RP4: SID
RP5: FNC
RP6: SIZ
RP7: Data 1
} Valid only when RP1 = 00

.

.

RPn+5: Data n

3) Return codes

RCD00

RCD12

RCD13

RCD14

4) Function

Receives data with a header from the serial port. FMT through SIZ are omitted if the header information is 01.

(5) ROM and speaker

This subsection deals with the commands for PROM capsules and speakers.

Table 3-5 PROM and speaker commands

Code	Command
70	Turn on/off PROM capsule power
71	Read PROM data
72	Turn on/off speaker power
73	Beep
74	Melody

(5-1) Turn On/Off PROM capsule power (70)

1) Send parameter

SP1: ØØ: OFF

Others: ON

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Turns on or off ROM capsule power according to the parameter specification.

5) Note:

It will take one second after a power-on before the ROM capsule is ready.

(5-2) Read Data (71)

1) Send parameter

SP1: Power on flag

(ØØ: OFF Others: ON)

SP2: Data address (high-order)

SP3: Data address (low-order)

SP4: Data count (Ø: 256, 1: 1, ... FF: 255)

2) Receive parameter

(Valid only when Return code = ØØ.)

RP1: Data 1

•

•

RPn: Data n

3) Return code

RCDØØ

4) Function

Reads the specified bytes of data starting at the specified data address. Cartridge #0 is selected when the MSB of the data address is 0. Cartridge #1 is selected when the MSB is 1. When the power off flag is set to ØØ, this command turns off ROM capsule power after reading the data. ROM capsule power must be turned on before this command is executed.

(5-3) Turn On or Off Speaker Power (72)

1) Send parameter

SP1: Power on/off switch

Bit 7 Ø: OFF 1: ON

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Turns on or off the power to the speaker amplifier.

(5-4) Beep (73)

1) Send parameter

SP1: Period (high-order)

SP2: Period (low-order)

SP3: Duration (in 100 msec units)

2) Receive parameter

None

3) Return code

RCD00

4) Function

Sounds the speaker at the specified frequency (reciprocal of period) for the specified length of time. The unit of period is 3.2 μsec.

(5-5) Melody (74)

1) Send parameter

SP1: Repeating count

(Ø: 256, 1: 1, ... FF: 255)

SP2: Data address (high-order)

SP3: Data address (low-order)

2) Receive parameter

None.

3) Return code

RCDØØ

4) Function

Sounds the speaker according to the data read from the specified address.

Data format

Duration 1 (1)

Period 1 (2)

•

•

Duration n (1)

Period n (2)

ØØ

Return Codes (in decimal)

1) RCDØØ (SYS)

Code: ØØ

Explanation: Normal termination.

2) RCDØ1 (SYS)

Code: Ø1

Explanation: Break acknowledged.

3) RCDØ2 (SYS)

Code: Ø2

Explanation: Command error.

A command code (ØØH - 7FH) not defined by the system was issued.

4) RCDØ3 (SYS)

Code: Ø3

Explanation: Communications error.

A command was issued while sending or receiving data or sending another command.

5) RCDØ4 (LCD)

Code: 11

Explanation: Invalid size specification.

The specified data did not fit in the screen.

6) RCD05 (LCD)

Code: 12

Explanation: Undefined graphics user defined character.

8) RCD06 (LCD)

Code: 13

Explanation: Invalid user defined character

An attempt was made to specify a code other than the codes under which user defined characters are defined.

9) RCD07 (MCT)

Code: 41

Explanation: Head error.

The head did not function normally.

10) RCD08 (MCT)

Code: 42

Explanation: Tape stopped during processing.

11) RCDØ9 (MCT)

Code: 43

Explanation: Write protect error.

An attempt was made to write on the tape having no write protect tab.

12) RCD1Ø (MCT)

Code: 44

Explanation: Data error.

A pulse of an invalid width was received and the logical state of the pulse (1 or Ø) could not be determined.

13) RCD11 (MCT)

Code: 45

Explanation: CRC error.

14) RCD12 (ESPS)

Code: 61

Explanation: Linking unsuccessful.

15) RCD13 (ESPS)

Code: 62

Explanation: Communication error.

An overrun framing error occurred.

16) RCD14 (ESPS)

Code: 63

Explanation: Time over.

17) RCD15 (BEEP)

Code: 71

Explanation:

A BEEP or MELODY command was issued before the execution of the preceding BEEP or MELODY command was completed.

18) RCD11-1 (MCT)

Code: 46

Explanation: Block mode error.

A block with an invalid block identifier was read.