<<Supported function list>> : Overseas version

This file is a supplement data at the library.
Don't use "INT" call from the program.

[Search one data]	AH: 00h	
[Read one data]	AH: 01h	
[Write one data]	AH: 02h	
[Correct one data]	AH: 04h	
[Delete one data]	AH: 05h	
[Delete mode data]	AH: 06h	
[Move one data]	AH: 07h	
[Notify the number of registered items]	AH: 08h	
[Notify the number of free blocks]	AH: 09h	
[Notify the total number of blocks of the integrated Flash]		AH: 0Ah
[Information service for pointer]	AH: 0Eh	
[Save system data onto the Flash.]	AH: 0Fh	
[Correct one data] (Only Schedule mode)	AH: 10h	
[Read one data]	AH: 11h	
[Delete mode data] (Only TODO mode.)	AH: 16h	
[Notify the number of registered items]	AH: 18h	
[Add Cradle ID information.]	AH: 20h	
[Get Cradle ID information.]	AH: 21h	
[Clear the correction bit.] (For Cradle)	AH: 22h	
[Memory remaking process]	AH: 30h	
[Get a pair pointer for Name/Company] (Only TEL mode)		AH: 40h
[Check registration area] (Only EXPENSE mode)		AH: 50h
[Get a next alarm.]	AH: 80h	
[Re-get a next alarm.]	AH: 81h	
[Convert pointer information]	AH: 82h	
[Get a next-next alarm.]	AH: 83h	
[Get a next alarm.]	AH: 84h	
(Exclude specified mode data.)		
[Read block data from Flash memory.]	AH: F0h	
[Write block data to Flash memory.]	AH: F1h	
[Erase sector of Flash memory.]	AH: F4h	
[Notify the device capacity of Flash memory.] (For backup.)		AH: F5h
[Notify the system status.]	AH: FAh	

♦ **Search one data**

[Input conditions]

<AX>Function number

AH: 00h

AL: Direction information

FFh: Initial data search

FEh: Final data search

00h: Next data search

01h: Previous data search

<BX>BH: Search condition

00h: No search condition

01h: Search by characters

02h: Initial letter search

04h: Account name search (Perfect match search)

08h: Search by time.

10h: Search by date.

20h: Cradle ID search

(Sets the search ID to <CX>.)

40h: Cradle correction bit search

<DX> Pointer (Data position during display.)

<DI> Top offset where the structure is placed.

<ES> Top segment where the structure is placed.

[Output conditions]

<AX>Error status

0000h: Normal end

Others: Further details will be informed.

<BX>BH: Registration data format

00h: Text

01h: Binary

<DX>Pointer (Top/next/previous data position)

[Usage]

Call after setting conditions to the following items of the structure:

- Mode entry number
- Subentry number
- Secret information

(Next two items are required when searching TODO data.)

- TODO check information
- TODO rank information

Buffers that are not corresponding will not be referred. So, it is not necessary to initialize those buffers. (When entering the mode, perform a search with AL=FFh.)

* When setting the common to the subentry number, the subentry number of the data is set when getting the data.

[Others]

- If the data searched is found in a period between the start date and the end date when performing data search by the term mode, then the data is regarded as a match. The date search by the common mode is made only by comparing the start date.
- When performing data search by the common (subentry: 00h) of the Schedule mode, the holiday setting information will not be found.
- When performing data search by the common (subentry: 00h) of the TEL mode, the owner registration information will not be found.

* Buffers that the structure requires are up to the search buffer. So, it is not necessary to prepare a buffer like [fsb_text_buf_] in the C structure.

♦ **Read one data**

[Input conditions]

<AX>Function number
AH: 01h
<BX>BH: Registration data format
00h: Text
01h: Binary
<DX>Pointer (Data position)
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Output conditions]

<AX>Error status
0000h: Normal end
Others: Further details will be informed.
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Usage]

Call after setting conditions to the following items of the structure:
- Mode entry number
- Subentry number

The buffers other than above are not referred. So, it is not necessary to initialize those buffers.

♦ **Write one data**

[Input conditions]

<AX>Function number

AH: 02h

<BX>BH: Registration data format

00h: Text

01h: Binary

<DI>Top offset where the structure is placed.

<ES>Top segment where the structure is placed.

[Output conditions]

<AX>Error status

0000h: Normal end

Others: Further details will be informed.

<DX>Pointer (Registration data position)

[Usage]

Call after setting conditions to the following items of the structure:

- Mode entry number
- Subentry number
- Secret information

(Followings are required when registering "Schedule" data.)

- Alarm check information

When an alarm is set, set data to the alarm date/time buffer.

(Followings are required when registering "TODO" data.)

- TODO check information
- TODO rank information

The buffers other than above are not referred. So, it is not necessary to initialize those buffers.

[Others]

- Sets the end date to the "Check date" in the structure.
- Both "(mode) 10h, (sub-mode) 02h" and "(mode) 20h, (sub-mode) 02h" registrations are proceeded as the Company TEL.

♦ **Correct one data**

[Input conditions]

<AX>Function number

AH: 04h (10h: Only Schedule mode)

<BX>BH: Registration data format

00h: Text

01h: Binary

<DX>Pointer (Data position before correction)

<DI>Top offset where the structure is placed.

<ES>Top segment where the structure is placed.

[Output conditions]

<AX>Error status

0000h: Normal end

Others: Further details will be informed.

<DX>Pointer (Data position after correction)

[Usage]

Call after setting conditions to the following items of the structure:

- Mode entry number
- Subentry number
- Secret information

(Followings are required when registering "Schedule" data.)

- Alarm check information

When an alarm is set, set data to the alarm date/time buffer.

(Followings are required when registering "TODO" data.)

- TODO check information
- TODO rank information

The buffers other than above are not referred. So, it is not necessary to initialize those buffers.

[Others]

- In the normal correction process, the correction bit is set unconditionally. However, in the correction during the cradle is processing, it causes a problem to set a correction bit. To avoid this, sets bit-7 (80h) of the BH input condition. This is regarded as a correction during the cradle is processing and makes it possible not to set a correction bit internally.

  [Input conditions] (Addition)

  <BX>BH: Registration data format

  00h: Text

  01h: Binary

  +

  80h: Cradle is processing.

- Both "(mode) 10h, (sub-mode) 02h" and "(mode) 20h, (sub-mode) 02h" corrections are proceeded as the Company TEL.

♦ **Delete one data**

[Input conditions]

<AX>Function number

AH: 05h

<DX>Pointer (Delete-data position)

[Output conditions]

<AX>Error status

0000h: Normal end

Others: Further details will be informed.

[Usage]

Gets the pointer for the previous (next) data using "data search" before issuing this command. After the deletion is complete, performs the data display based on the pointer.

♦ **Delete mode data**

[Input conditions]

<AX>Function number

AH: 06h

<DI>Top offset where the structure is placed.

<ES>Top segment where the structure is placed.

[Output conditions]

<AX>Error status

0000h: Normal end

Others: Further details will be informed.

[Usage]

Call after setting conditions to the following items of the structure:

- Mode entry number
- Subentry number
- Secret information

The valid numeric number as sub-entry is a single mode information or 00h.

* Only three items (listed above) in the structure are required. Therefore, it is not necessary to allocate other area.

♦ **Move one data**

[Input conditions]

<AX>Function number

AH: 07h

<BX>Pointer (Move-To data position)

<DX>Pointer (Move-data position)

[Output conditions]

<AX>Error status

0000h: Normal end

Others: Further details will be informed.

<DX>Pointer (Move-data data position after moved)

♦ **Notify the number of registered items**

[Input conditions]

<AX>Function number

AH: 08h

<DI>Top offset where the structure is placed.

<ES>Top segment where the structure is placed.

[Output conditions]

<AX>Error status

0000h: Normal end

<CX>Number of records

[Usage]

Call after setting conditions to the following items of the structure:

- Mode entry number
- Subentry number
- Secret information

(Following two items are required when searching TODO data.)

- TODO check information
- TODO rank information

The buffers other than above are not referred. So, it is not necessary to initialize those buffers.

* Only five items (listed above) in the structure are required. Therefore, it is not necessary to allocate other area.

♦ **Notify the number of free blocks**

[Input conditions]

<AX>Function number

AH: 09h

[Output conditions]

<AX>Error status

0000h: Normal end
<CX>Free block numbers

♦ **Notify the total number of blocks of the integrated Flash**

[Input conditions]
<AX>Function number
AH: 0Ah

[Output conditions]
<AX>Error status
0000h: Normal end
<DX>Total block numbers of integrated Flash.

♦ **Information service for pointer**

[Input conditions]
<AX>Function number
AH: 0Eh
<DX>Pointer
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Output conditions]
<AX>Error status
0000h: Normal end
Others: Further details will be informed.
<BX>BH: Registration data format
00h: Text
01h: Binary
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Usage]
Call after setting conditions to the following items of the structure:
- Mode entry number
- Subentry number
- Secret information

* Only three items (listed above) in the structure are required. Therefore, it is not necessary to allocate other area.

♦ **Push the system data onto the Flash.**

[Input conditions]
<AX>Function number
AH: 0Fh

AL: Registration position
01h: Save BIOS system area.
02h: Save C system area.

[Output conditions]
<AX>Error status
0000h: Normal end
Others: Further details will be informed.

♦**Read one data**
[Input conditions]
<AX>Function number
AH: 11h
<BX>BH: Registration data format
00h: Text
01h: Binary
<CX>Block numbers to read out.
<DX>Pointer (Data position)
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Output conditions]
<AX>Error status
0000h: Normal end
Others: Further details will be informed.
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

♦ **Delete mode data** (ONLY TODO MODE.)
[Input conditions]
<AX>Function number
AH: 16h
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Output conditions]
<AX>Error status
0000h: Normal end
Others: Further details will be informed.

[Usage]
Call after setting conditions to the following items of the structure:
- Mode entry number : 40h fixed.
- Subentry number : 08h fixed.
- Secret information

- TODO check information : Deleting condition (FFh valid)
- TODO rank information : Deleting condition (FFh valid)

The buffers other than above are not referred. So, it is not necessary to initialize those buffers.

[Purpose]

This is used to delete the data in the TODO mode (categories are ignored).
The data deletion for each category in the mode should be performed using AH = 06h.

♦ **Notify the number of registered items**

[Input conditions]

<AX>Function number
AH: 18h
<BX>BH: Registration data format
00h: Text
01h: Binary
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Output conditions]

<AX>Error status
0000h: Normal end
<CX>Number of records

[Usage]

Call after setting conditions to the following items of the structure:

- Mode entry number
- Subentry number
- Secret information

(Followings are required when searching TODO data.)

- TODO check information
- TODO rank information

The buffers other than above are not referred. So, it is not necessary to initialize those buffers.

* Only five items (listed above) in the structure are required. Therefore, it is not necessary to allocate other area.

♦ **Add Cradle ID information**

[Input conditions]

<AX>Function number
AH: 20h
<CX>Cradle ID
<DX>Pointer (Data position)

[Output conditions]

<AX>Error status

0000h: Normal end
Others: Further details will be informed.

♦ **Get Cradle ID information**

[Input conditions]
<AX>Function number
AH: 21h
<DX>Pointer (Data position)

[Output conditions]
<AX>Error status
0000h: Normal end
Others: Further details will be informed.
<CX>Cradle ID

♦ **Clear the correction bit** (For Cradle)

[Input conditions]
<AX>Function number
AH: 22h
<DX>Pointer (Data position)

[Output conditions]
<AX>Error status
0000h: Normal end
Others: Further details will be informed.

♦ **Memory rearrangement process**

[Input conditions]
<AX>Function number
AH: 30h

[Output conditions]
<AX>Error status
0000h: Normal end

♦ **Get a pair pointer for Name/Company** (ONLY TEL MODE)

[Input conditions]
<AX>Function number
AH: 40h
<DX>Pointer (Limited on TEL mode)

[Output conditions]

<AX>Error status
0000h: Normal end
<BX>Personal pointer
<DX>Company pointer (FFFFh: Not paired.)
Cannot register as Company if the pointer is not paired.

[Purposes]
This is used to get one of a pair of pointers for the name sort or company sort from the data registered to TEL-Company. Use this for the history function of the telephone.

♦ **Check registration area** (ONLY EXPENSE MODE)
[Input conditions]
<AX>Function number
AH: 50h

AL: 00h Specify initialization
: 01h Execute area check

<BX>Registration data format
BH: 00h Text
: 01h Binary

<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Output conditions]
<AX>Error status
0000h: Normal end
0008h: Memory full
FFFEh: Parameter error

[Process]
When specifying the initialization, writes the current number of registered items and the number of free blocks in each user sector to the buffer, and virtually rewrites the buffer to make it possible for performing the memory full check and others when executing the area check.
Both the number of items and the number of free blocks are checked for the memory full.

♦ **Get a next alarm**
[Input conditions]
<AX>Function number
AH: 80h
<DI>Top offset where alarm structure is placed.
<ES>Top segment where alarm structure is placed.

[Process]

After clearing the alarm information of data that is indicated by "fsb_alam_buf_" pointer, searches next alarm information and returns that information to "fsb_alam_buf_".

♦ **Re-get a next alarm**

[Input conditions]

<AX>Function number

AH: 81h

<DI>Top offset where alarm structure is placed.

<ES>Top segment where alarm structure is placed.

[Process]

Calls this if it is expected that the next alarm is affected when performing data registration, deletion or correction. At this time, do not delete the alarm bit on the Flash memory for the data specified by "fsb_alam_buf_" pointer.

♦ **Convert pointer information**

[Input conditions]

<AX>Function number

AH: 82h

<DI>Top offset where alarm structure is placed.

<ES>Top segment where alarm structure is placed.

[Output conditions]

DX: RELATIVE POINTER FOR C (FOR DISPLAY)

AX: 0000h (normal)/others (no next alarm)

♦ **Get a next-next alarm**

[Input conditions]

<AX>Function number

AH: 83h

<DI>Top offset where alarm structure is placed.

<ES>Top segment where alarm structure is placed.

[Process]

Searches a next alarm data of the alarm data specified by "fsb_alam_buf_" pointer, and updates the "fsb_alam_buf_" pointer.

♦ **Get a next alarm** (Excluding specified mode data.)

[Input conditions]

<AX>Function number

AH: 84h

<BX>Not look at alarm time, mode information.
BH: Subentry number
BL: Secret information
<DI>Top offset where alarm structure is placed.
<ES>Top segment where alarm structure is placed.

[Process]
Gets next alarm data except data to be deleted in the all mode deletion. The information to set to the structure is equivalent to the conditions of the mode data deletion.

♦ **Read block data from Flash memory.**
[Input conditions]
<AX>Function number
AH: F0h
<BX>BH: Sector position to read out.
00h - 3Fh

<CX>Byte numbers to read out. (Specify by even byte.)
<DX>Offset address to read out. (vs. Flash)
<ES: DI>Data position to read out. (ON SRAM)

[Output conditions]
<AX>Error status
0000h: Normal end
FFFEh: Input parameter error.

♦ **Write block data to Flash memory**
[Input conditions]
<AX>Function number
AH: F1h
<BX>BH: Sector position to be written. (64-Kbyte units)
00h - 3Fh
<CX>Byte numbers to write. (Specify by even byte.)
<DX>Offset address to write (vs. Flash)
<ES: DI>Data position to be written. (ON SRAM)

[Output conditions]
<AX>Error status
0000h: Normal end
FFFEh: Input parameter error.

♦ **Erase sector of Flash memory**
[Input conditions]

<AX>Function number
AH: F4h
<BX>BH: Sector position to delete. (64-Kbyte units)
00h - 3Fh

[Output conditions]
<AX>Error status
0000h: Normal end
FFFEh: Input parameter error.
FFFFh: Fatal error (Cannot read/write FLASH.)

[Others]
- The break key sample check is included.

♦ **Notify the device capacity of Flash memory**

[Input conditions]
<AX>Function number
AH: F5h
AL: Flash to read out.
00h: Total of all Flash integrated.
01h: ONLY USER FLASH1.
02h: ONLY USER FLASH2.

[Output conditions]
<AX>Error status
0000h: Normal end
FFFEh: Input parameter error.

<DX>Capacity of Flash memory (in Kbytes)
0, 1024, 2048, 3072, 4096

♦ **Notify the system status**

[Input conditions]
<AX>Function number
AH: FAh

[Output conditions]
<AX>Error status
0000h: Normal end
FFFFh: Data error is occurring!

♦ **Structure**

Build the following structures in the C area.

[Text edition]

fsb_main_entry_ (1 byte)	: Mode entry number
fsb_sub_entry_ (1 byte)	: Subentry number
fsb_scrt_info_ (1 byte)	: Secret information
fsb_ararm_chk_ (1 byte)	: Alarm check information
fsb_todo_chek_ (1 byte)	: TODO check information
fsb_todo_rank_ (1 byte)	: TODO rank information
fsb_date1_buf_ (8 bytes)	: Date/Due date
fsb_date2_buf_ (8 bytes)	: Check date
fsb_date_aram_ (8 bytes)	: Alarm date
fsb_time1_buf_ (4 bytes)	: Start time/Due time (from)
fsb_time2_buf_ (4 bytes)	: End time/Due time (to)
fsb_time3_buf_ (4 bytes)	: Check time
fsb_time_aram_ (4 bytes)	: Alarm time
fsb_date_srch_ (8 bytes)	: Date search buffer
fsb_time_srch_ (4 bytes)	: Time search buffer
fsb_text_srch_ (33 bytes)	: Real data (text) search buffer. [Common for read/real data]
fsb_text_buf_ (2049 bytes)	: Buffer to store real data (text).

[Binary edition]

fsb_main_entry_ (1 byte)	: Mode entry number
fsb_sub_entry_ (1 byte)	: Subentry number
fsb_scrt_info_ (1 byte)	: Secret information
fsb_ararm_chk_ (1 byte)	: Alarm check information
fsb_todo_chek_ (1 byte)	: TODO check information
fsb_todo_rank_ (1 byte)	: TODO rank information
fsb_date1_buf_ (8 bytes)	: Date/Due date
fsb_date2_buf_ (8 bytes)	: Check date
fsb_date_aram_ (8 bytes)	: Alarm date
fsb_time1_buf_ (4 bytes)	: Start time/Due time (from)
fsb_time2_buf_ (4 bytes)	: End time/Due time (to)
fsb_time3_buf_ (4 bytes)	: Check time
fsb_time_aram_ (4 bytes)	: Alarm time
fsb_date_srch_ (8 bytes)	: Date search buffer
fsb_time_srch_ (4 bytes)	: Time search buffer

fsb_text_srch_ (33 bytes) : (Dummy)

fsb_dummy_22by_ (22 bytes) : Used for memory management when performing registration.

fsb_char_num_ (2 bytes) : Binary data capacity
fsb_bin_buf_ (9600 bytes) : Buffer to store real data (binary).

♦ **Buffer**

【Binary edition】

fsb_bin_buf_ (9600 bytes) : Buffer to store real data (binary).
fsb_bin_tsize_ (2 bytes) : Total byte numbers

Deleted: Think this by each application side.
fsb_bin_xsize_ (2 bytes) : X coordinate (horizontal) dot numbers fsb_bin_ysize_ (2 bytes) : Y coordinate (vertical) dot numbers **<Take this into your consideration>** 1. One screen buffer is available. However, it is not expected to use full of the buffer. 2. X-coordinate should always be an 8 multiple. 3. The total number of bytes can be calculated as follow: ("Number of dots for X-coordinate" ÷ 8) **x** "Number of dots for Y-coordinate" It is necessary to set the total number of bytes before the registration.

fsb_dummy_22by_ (22 bytes) : Used for memory management when performing the registration.

【Text edition】

fsb_text_buf_ (2049 bytes) : Buffer to store real data (text).
fsb_text_srch_ (33 bytes) : Real data (text) search buffer.

<Take this into your consideration>

(1) Insert an END code (00h) to the tail of the text.
(Name read/Company read are as well.)

【Date/Day and Time】

fsb_date1_buf_ (8 bytes) : Date/Due date
fsb_date2_buf_ (8 bytes) : Check date
fsb_date_aram_ (8 bytes) : Alarm date
fsb_date_srch_ (8 bytes) : Date search buffer
fsb_time1_buf_ (4 bytes) : Start time/Due time (from)
fsb_time2_buf_ (4 bytes) : End time/Due time (to)
fsb_time3_buf_ (4 bytes) : Check time
fsb_time_aram_ (4 bytes) : Alarm time
fsb_time_srch_ (4 bytes) : Time search buffer

<Take this into your consideration>

(1) Date format

[YYYYMMDD]

YYYY: Shows year. Remove a separation code and show "0" by "30h".

MM: Shows month. Remove a separation code and show "1" by "31h".

DD: Shows day. Remove a separation code and show "2" by "32h".

(2)Time format

[HHMM]

HH: Shows hour. Remove a separation code and show "3" by "33h".

MM: Shows minute. Remove a separation code and show "4" by "34h".

(3)Result

Possible values are "0" to "9". That is, 30h to 39h are possible values.

(4)Clear time data by "00h" for the data without time.

【Others】

fsb_main_entry_ (1 byte) : Mode entry number
See [Data Structure].

fsb_sub_entry_ (1 byte) : Subentry number
See [Data Structure].

fsb_scrt_info_ (1 byte) : Secret information
80h: None
00h: Present

fsb_todo_chek_ (1 byte) : TODO check information
04h: None
00h: Present
FFh: Ignore item(s) (Using for search.)

fsb_todo_rank_ (1 byte) : TODO rank information
03h: Rank A
01h: Rank B
00h: Rank C
FFh: Ignore item(s) (Using for search.)

---

fsb_ararm_chk_ (1 byte) : Alarm check information
40h: Present
00h: None

♦ **About Search Method**

(1) Search by characters

All character string data are searched. Therefore, if the search string is found anywhere in the sentence, it means that the search data is hit.

(2) Initial letter search

Targets only the top string of the top item.

(3) Search by time.

If data that dose not employ a term (except Schedule) is searched, only a match or no match can be a search result.

If the data that employs a term (Schedule) is searched, the search result follows the rules described below.

Schedule term (period) : from AM 9: 00 to AM 11: 00

Search data :	AM 8: 30	- - - Unmatched
	AM 9: 00	- - - Matched
	AM 10: 00	- - - Matched
	AM 11: 00	- - - Unmatched
	AM 11: 30	- - - Unmatched

(4) Search by date.

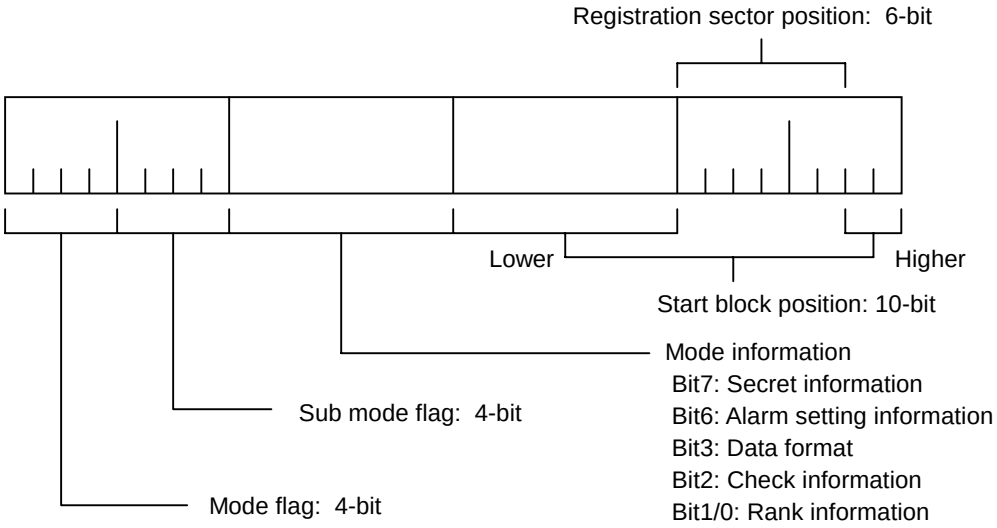
Whether the specified date is simply hit or not.

♦ **Data Structure** (Attached documents.)

【Common】 : Data structure in the control sector

4-byte data (See [Data structure in the control sector].) are gathered.

The 4-byte data are classified by mode, sorted by a specific sort condition of each mode, and then stored.



[Mode entry]

10h: TEL
20h: TEL (Company)
30h: Memo
40h: Schedule
50h: Spreadsheet
60h: Mail
70h: EXPENSE
80h: Clock (City setting)
90h: Dual window
A0h: ADD IN
B0h: Quick Memo

[Sub-mode entry number]

<TEL>
- 00h: Common
- 01h: Personal
- 02h: Company
- 03h: Untitled1
- 04h: Untitled2
- 05h: Untitled3
- 06h: Untitled4
- 07h: Untitled5
- 0Fh: Owner

<Memo>
- 00h: Common
- 01h: Untitled1
- 02h: Untitled2
- 03h: Untitled3
- 04h: Untitled4
- 05h: Untitled5

<Schedule>
- 01h: Schedule
- 02h: Schedule (Term registration)
- 03h: Reminder
- 04h: Holiday setting
- 06h: Schedule (Common)
- 08h: TODO (Common)
- 09h: TODO (CATEGORY A)
- 0Ah: TODO (CATEGORY B)
- 0Bh: TODO (CATEGORY C)
- 0Ch: TODO (CATEGORY D)
- 0Dh: TODO (CATEGORY E)
- 0Eh: TODO (CATEGORY E)
- 0Fh: TODO (CATEGORY F)
- 00h: Common

<EXPENSE>
- 01h: Account
- 02h: Transaction
- 03h: Payment type

<Dual-window>
- 01h: Clip

[Mode information]

Bit 7: Secret information
- 1: OFF
- 0: ON

Bit 6: Alarm setting information
- 1: Present
- 0: None

Bit 3: Data format
- 1: Binary
- 0: Text

Bit 2: Check information
- 1: None
- 0: Present

Bit 1/0: Rank information
- 11: Rank A
- 01: Rank B
- 00: Rank C

Note

"08h" cannot be add to the sub mode entry value. "08h" is used to perform the special process only for TODO mode.

* Schedule (Common) <06h> can be used only for search. This is effective to search Schedule/Schedule (term) at once.

* When calling Schedule common item, the holiday setting is not a target of the call.

<Quick Memo>
- 00h: Common
- 01h: Untitled1
- 02h: Untitled2
- 03h: Untitled3

【Individual modes】 : Data information for an item.

"FT" (Flag to specify the start of mode.) and "RT" (Flag to specify the start of one data.) should not be used in this system since the data of this system are not registered sequentially while the ordinal digital diaries use such sequential data registration.

However, "IT" (Flag to specify the start of each item) is still used as "FEh" value.

[TEL]

2	1~	1	1~
Cradle ID	Name	IT	Content

[Memo]

2	2~
Cradle ID	Content

[Schedule/TODO/Reminder]

<Schedule>

2	4	2	2	4	2	1~
Cradle ID	Date	Start time	End time	Alarm date	Alarm time	Content

<Schedule(Term)>

2	4	2	2	4	2	4	2~
Cradle ID	Start date	Start time	End time	Alarm date	Alarm time	End date	Content

<TODO>

2	4	2	2	4	2	4	2	1~
Cradle ID	Due date	Due time (from)	Due time (to)	Alarm date	Alarm time	Check date	Check time	Content

<Reminder>

2	4	2	2	4	2	1~
Cradle ID	Due date	Due time (from)	Due time (to)	Alarm date	Alarm time	Content

<Binary> : Holiday settings

2	4	
Cradle ID	Date	Information for 1 year (1 byte for a week) is held.

[Spreadsheet]

2	1~
Cradle ID	Content

[Mail]

2	1~
Cradle ID	Content

[Quick Memo]

2	1~
Cradle ID	Content

[EXPENSE]

<Account>

2	1~	1	1~
Cradle ID	Account name	IT	Content

<Transaction>

2	4	1~	1	1~
Cradle ID	Date	Account name	IT	Content

<Payment type>

2	1~
Cradle ID	Content

<<Usage>>

When registering <transaction>, set the account name to the area that is equivalent to "fsb_text_srch_" in the "structure".

【Factors in the data structure】

[Date (Year-Month-Day/Due date/Check date]

Year	Month	Day

- Year: A value ranging from 1901 to 2099 in BCD.
- Month: A value ranging from 01 to 12 in BCD.
- Day: A value ranging from 01 to 31 in BCD.

[Time (Due time/Alarm time/heck time/Start time/End time)]

Hour	Minute

- Expresses the value in 24-hour system in BCD.
  However, the value for "Hour" is stored after adding "1" to the high-order side of "Hour".
- All "0" indicate that there is no data.
- Example) "AM 5: 30" becomes "1530h".
  "PM 9: 45" becomes "3145h".

Makes slight modification on the initial data search in the "**Search one data**" so that the one month display can be performed easily.
The contents of the function are as follows:

♦ **Search one data**
[Input conditions]
<AX>Function number
AH: 00h
AL: Direction information
FFh: Initial data search
<BX>BH: Search condition
10h: Search by date
<DI>Top offset where the structure is placed.
<ES>Top segment where the structure is placed.

[Output conditions]
<AX>Error status
0000h: Normal end
4444h: Nearest data (Most likely)
Others: Further details will be informed. (This is an error.)
<BX>BH: Registration data format
00h: Text
01h: Binary
<DX>Pointer (Top data position)

【Programming method】
When entering the start date of the month (for instance, 3/1 for March) during one month display, the nearest one to "3/1" is picked up. (AX=0000h does not cause a problem..)
At this time, if AX=4444h, then perform a read operation by that pointer, and check if the date is included in the target "month".

In the subsequent data read, perform the search operation with no search condition when there is no specific search condition, and then perform a read operation by that pointer. After checking if the date is included in the target "month", perform individual processes.

♦ **Alarm**
When the factor to change the alarm time is occurred, the alarm time update process must be performed in each application.

The following factors are to change the alarm time:
1. Daily alarm change. (Includes the status change <ON/OFF> and so on.)
2. Process for the Schedule data with alarm. (Registration/Delete/Change/Move process.)
3. Next alarm process after the alarm sounded.

<<Alarm setting >>

1. Daily alarm change (This also occurs when changing the city setting.)
   (1) If the daily alarm sound setting is valid, get the appropriate alarm time using "Get daily alarm" BIOS.
   (2) If the Schedule alarm sound setting is valid, get the appropriate alarm using "Re-get a next alarm" BIOS.
   (3) If the alarm data obtained at above step (2) is already old, then repeat "Get a next alarm" until near future alarm is found.
   (4) Compares two alarm times obtained above (1) and (3), and a near future time should be set as a next alarm.

2. Process for the Schedule data with alarm.
   There are several ways to achieve this process. So, following examples are just for your reference.

   TYPE A) Registration process
   For the data with alarm, the same settings described in "1. Daily alarm change " can be applied.

   TYPE B) Delete process
   (1) Checks if the deleting data and the next alarm setting data match.
   (2) If they don't match, delete data without alarm process.
   If they match, perform following processes.
   1. If the daily alarm sound setting is valid, get the appropriate alarm time using "Get a daily alarm" BIOS.
   2. If the Schedule alarm sound setting is valid, get the appropriate alarm using "Get a next-next alarm" BIOS.
   3. Compares two alarm times obtained above 1. and 2., and a near future time should be set as a next alarm.

   TYPE C) Correction/Move process
   (1) Check if the alarm data exists in the data before correction.
   No alarm data found: No preprocessing required.
   Alarm data found: Perform the preprocessing.
   (2) Check if the alarm data exists in the data after correction.
   No alarm data found: No post-processing required.
   Alarm data found: Perform the post-processing.
   About "preprocessing": This means "TYPE B) Delete process".
   About "post-processing": This means "TYPE A) Registration process".

   TYPE D) All data delete process
   (1) Checks if the alarm setting data is included in the data to delete.
   (2) If they don't match, delete data without processing alarm.
   If they match, perform following processes.
   1. If the daily alarm sound setting is valid, get the appropriate alarm time using "Get a daily alarm" BIOS.
   2. If the Schedule alarm sound setting is valid, get the appropriate alarm using "Get

a next alarm" (excluding the mode data specified).

3. Compares two alarm times obtained above 1. and 2., and a near future time should be set as a next alarm.

3. Next alarm process after the alarm sounded.
   (1) If the daily alarm sound setting is valid, get the appropriate alarm time using "Get a daily alarm" BIOS.
   (2) If the Schedule alarm sound setting is valid, get the appropriate alarm using "Re-get a next alarm" .
   (3) If the alarm data obtained at above step (2) is already old, then repeat "Get a next alarm" until near future alarm is found.
   (4) Compares two alarm times obtained above (1) and (3), and a near future time should be set as a next alarm.

<<Alarm process>>
   (1) Get the alarm information.
   (2) If it is the daily alarm, then follow the specification.
   (3) Change the pointer to be displayed from the absolute pointer to the relative pointer.
   (4) Display using that pointer.
   (5) After completing the display, don't go to the key wait process, but perform a next alarm process. (See "3. Next alarm process after the alarm sounded." above.)
   (6) Wait releasing of the alarm display using the key wait.

   * Next alarm display during the alarm display.
   A new data display is performed after closing the previous display since the alarm display is performed using the popup display.

♦ **Next alarm process**

【Buffer】 (BIOS side buffer)
The buffer described below is used to transfer the alarm information.
At the initialization, put "FFFFh" to the pointer to indicate that there is no alarm value. In addition, the pointer returns "FFFFh" as the return value when there is no alarm data.

```
fsb_alarm_buf_    DB    2 DUP (?)    ;Pointer * 1
                  DB    4 DUP (?)    :Alarm date
                  DB    2 DUP (?)    ;Alarm time
                  DB    1 DUP (?)    ;Mode information
                  DB    1 DUP (?)    ;Sub-mode information
                  DB    1 DUP (?)    :Registration data format
                  DB    1 DUP (?)    :Secret information
```

* 1: Pointer
The pointer described above is not the same pointer with a pointer used by C. (If you attempt to read data using this pointer, no correct information is returned.)

This pointer points the absolute address of the data with alarm. So, the conversion is necessary to use this pointer as a pointer for C.

♦ **Error function list**

```
IW_NORMALCY       :0000h    ;Normal.
IW_NUL_BUFF       :0001h    ;No data to register.
IW_WRITE_ERR      :0002h    ;X (Used only for internal error status.)
IW_FORMAT_ERR     :0003h    ;Fatal error (System crash is being generated.)
IW_MAX_CHAR_ERR   :0004h    ;The number of characters to be registered exceed the limit.
IW_TME_DATE_ERR   :0005h    ;The data other than numeric number exists. (Date/Time)
IW_OUT_PNTA_ERR   :0006h    ;The setting pointer has an illegal value (out of range).
IW_TWIN_DAT_ERR   :0007h    ;When deleting data in TEL<COMPANY> mode,
                            ;  there is no 4-byte data to be paired.
IW_MEMORY_OVER    :0008h    ;Unable to register because of the insufficient memory capacity.
IW_READ_DT_ERR    :0009h    ;Read buffer is empty. <<Domestic only>>: At registration.
IW_ARM_TOTL_ERR   :0010h    ;The number of registered alarms exceed the maximum.
IW_ARM_SAMV_ERR   :0011h    ;The same alarm value already exists.
IW_ARM_NULL_ERR   :0012h    ;Alarm is empty.
                  :0020h    ;The cradle ID of the write destination is not FFFFh.
IW_PARAMET_ERR    :0FFFEh   ;Illegal value is set: Debug purpose only.
IW_NO_DATA_ERR    :0FFFFh   ;No data exists for search.
```