4.3 Hooks

4.3.1 General

The PINE is provided with some hooks to facilitate the application program to extend the capabilities of the interrupt and BIOS functions. The PINE hooks are:

1. Alarm hook
2. Interrupt hook
3. BIOS hook

This section describes and show the use of the PINE hooks.

4.3.2 Hook Structures

4.3.2.1 Hook jump table

The hook jump table is located between ØFFCØH and ØFFEFH. Its structure is shown below.

FFCOH	ALMHKO	JP EF1FH
FFC3H	ALMHK1	JP EF1FH
FFC6H	ALMHK2	JP EF1FH
FFC9H	ALMHK3	JP EF1FH
FFCCH	ALMHK4	JP EF1FH
FFCFH	RESERVED	JP EF1FH
FFD2H	HK8251	JP EF1FH
FFD5H	ICFHOOK	JP EF1FH
FFD8H	OVFHOOK	JP EF1FH
FFDBH	EXTHOOK	JP EF1FH
FFDEH	TMDT83	JP EF1FH
FFE1H	TMDT85	JP EF1FH
FFE4H	TMDT86	JP EF1FH
FFE7H	BIOSHK	JP EF1FH
FFEAH	RESERVED	JP EF1FH
FFEDH	RESERVED	JP EF1FH

ØEF1F contains a RET instruction.

4.3.2.2 Hook types

The PINE hooks are divided into the following types:

(1) Alarm/wake hooks (5)

```
    ALMHKØ     (Alarm hook Ø)
    ALMHK1     (Alarm hook 1)
    ALMHK2     (Alarm hook 2)
    ALMHK3     (Alarm hook 3)
    ALMHK4     (Alarm hook 4)
```

(2) Interrupt hooks (4)

```
    HK8251     (RxRDY interrupt hook)
    ICFHOOK    (ICF interrupt hook)
    OVFHOOK    (OVF interrupt hook)
    EXTHOOK    (EXT interrupt hook)
```

(3) BIOS hooks (4)

```
    TMDT83 (TIMDAT hook)
    TMDT85 (TIMDAT hook)
    TMDT86 (TIMDAT hook)
    BIOSHK (BIOS hook)
```

(4) Reserved (3)

HK8251 is not supported by OS version 1.Ø.

4.3.2.3 Hook structure

The PINE executes its program modules while switching four banks. Depending on which bank the PINE is currently running, the hook processing mode is divided into the following two types:

1. Hooks in which control is transferred to the hook address after bank Ø (RAM) has been selected.
2. Hooks in which control is transferred to the hook address without switching the active bank (system bank).

(1) Hooks which run without switching to bank Ø (RAM)

The active bank is unpredictable when an interrupt occurs. For this reason, interrupt hook processing proceeds as follows:

1. Switch to bank Ø (RAM).
2. Cause the CPU to jump to the specified hook address.
3. Switch to the original bank.

Hooks which run in this mode include OVFHOOK, ICFHOOK, and EXTHOOK.

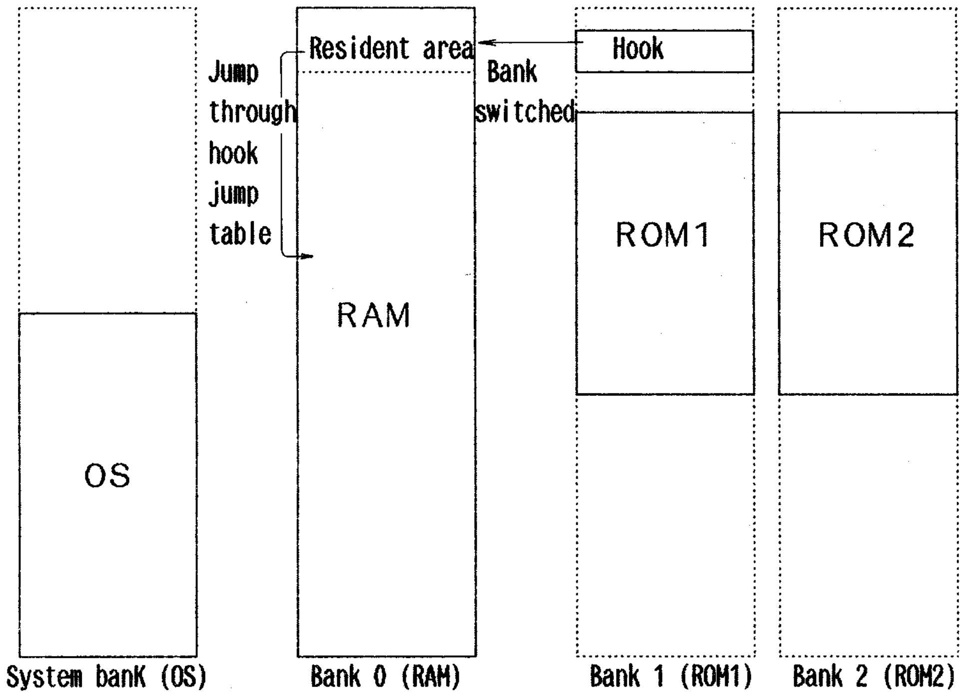


Fig. 4.3.1 Hooks Which Switch Banks

(2) Hooks which run without switching banks.
Hooks which are entered during execution of a module on OS ROM do not switch banks. When rewriting this type of hook, therefore, it is necessary to specify an address not lower than 8000H. Hooks of this type include: ALMHK0, ALMHK1, ALMHK2, ALMHK3, ALMHK4, HK8251, TMDT83, TMDT85, TMDT86, and BIOSHK.

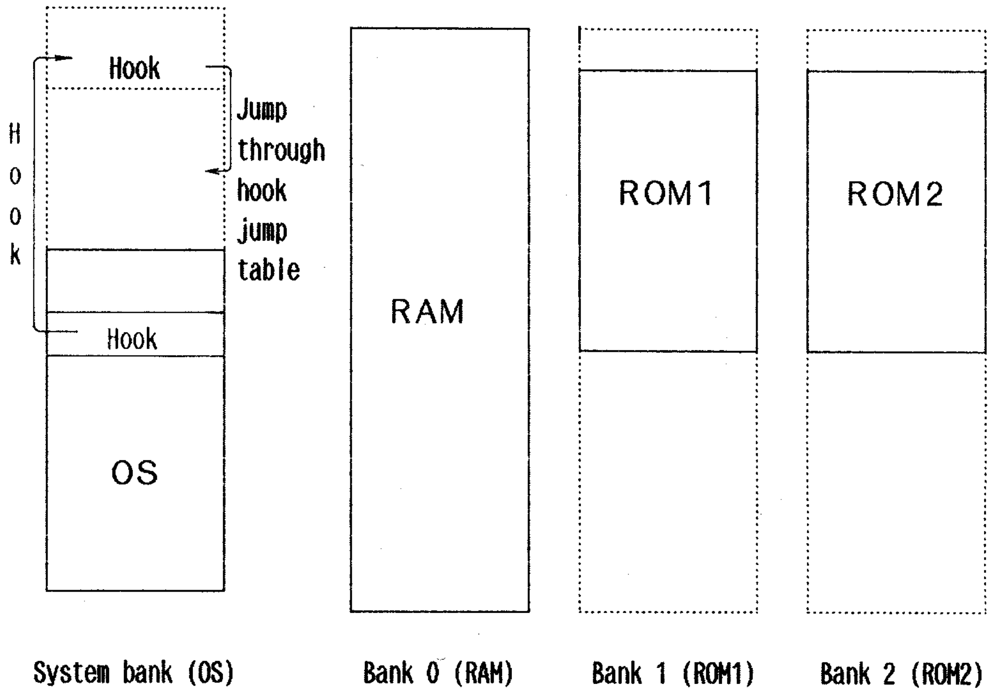


Fig. 4.3.2 Hooks Which Do Not Switch Banks

4.3.2.4 Programming notes

Since hooks are embedded in interrupt and BIOS processing modules, the following programming notes must be taken:

(1) Do not call BIOS/BDOS from a hook.
BIOS and BDOS establish their own stack area at the beginning of execution. Any call to BIOS or BDOS from a hook may destroy the stack area, causing a program hangup. To use a BIOS or BDOS function from a hook, call directly the BIOS or BDOS function on the OS ROM.

(2) Set up a stack area.
A stack overflow condition may occur if the called hook uses the stack established by the calling module. To prevent stack overflows, the hook processing modules must establish their own stack area. The stack area for hooks which do not switch banks must be located in a RAM area not lower than 8000H.

(3) Registers must be saved by the calling module.
The modules that receive control from a hook save the contents of only the required registers. The hook processing modules, therefore, must save all registers that are to be used during their execution. Failure to take this measure when implementing ICFHOOK or EXTHOOK may cause a program hangup on return from the interrupt processing module.

(4) Maintain the interrupt state.
Some interrupt modules run in the DI state and prevent subsequent interrupts. Accordingly, the hook processing modules must keep track of the interrupt state by themselves. The interrupt state when hooks are entered are as follows:

ALMHKØ	DI state
ALMHK1	DI state
ALMHK2	EI state, 75Ø8 disabled
ALMHK3	EI state, 75Ø8 disabled
ALMHK4	EI state
HK8251	DI state
ICFHOOK	DI state
OVFHOOK	DI state
EXTHOOK	DI state
TMDT83	EI state, 75Ø8 disabled
TMDT85	EI state, 75Ø8 disabled
TMDT86	EI state, 75Ø8 disabled
BIOSHK	EI state

(5) Disable interrupts when rewriting a hook.
Disable interrupts when rewriting the hook jump table. Particularly, be sure to disable interrupts when rewriting address data, one byte at a time.
(6)
The hook jump table is initialized during system initialize and reset.

4.3.3 Alarm Hooks

The PINE has five hooks for alarm processing. These hooks are used for automatic alarm/wake data update processing. The locations of the hooks are listed below.

Alarm hook Ø: Within the module that is activated when an alarm occurs in the power-off state and immediately before the alarm screen is displayed.
Alarm hook 1: Within the module that is activated when an alarm occurs in the power-off state and immediately after the alarm screen disappears.
Alarm hook 2: Within the module that is activated when an alarm occurs in the power-on state and immediately before the alarm screen is displayed.
Alarm hook 3: Within the module that is activated when an alarm occurs in the power-on state and immediately after the alarm screen disappears.
Alarm hook 4: Within the module that is activated when an alarm occurs and immediately before the module is exited due to a condition such as the depression of the ESC key.

The relationships between the alarm processing modules are shown in Figures. 4.3.3 through 4.3.5.

The alarm/wake update processing module, for example, sets the next alarm or wake time using alarm hook Ø and alarm hook 2 when an alarm or wake condition occurs.

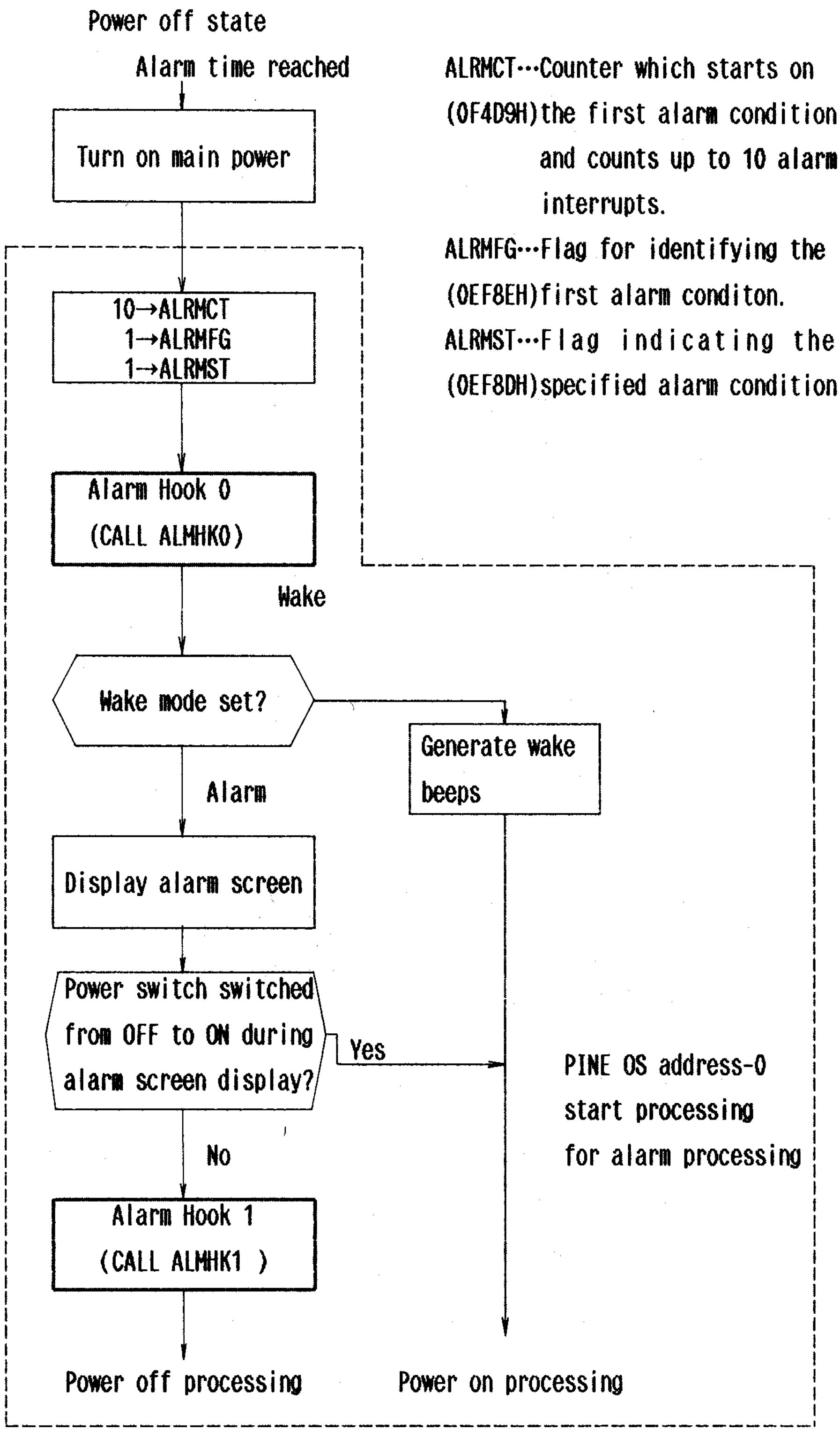


Fig. 4.3.3 Alarm Processing in the Power-off State

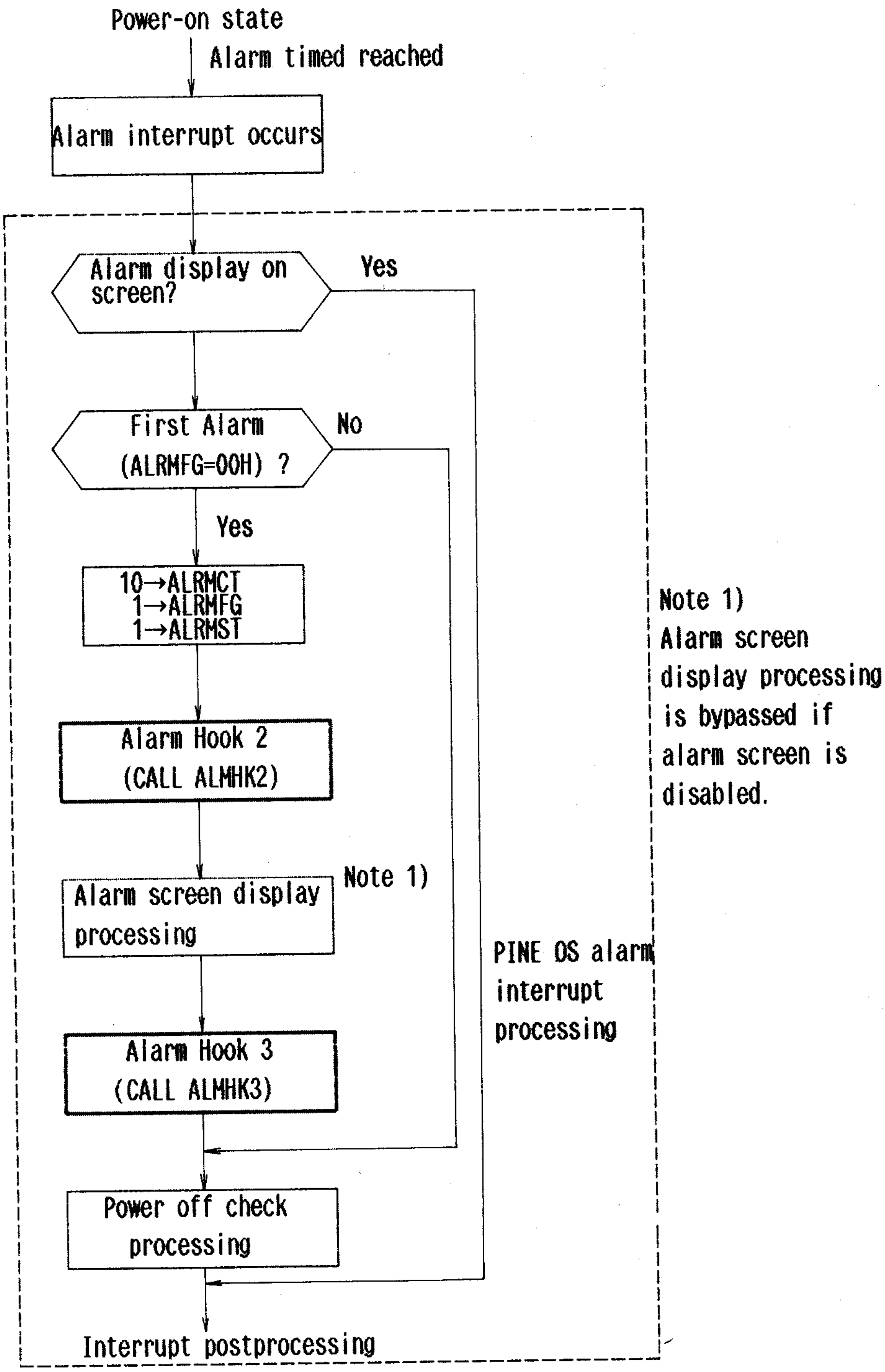


Fig. 4.3.4 Alarm Processing in the Power-on State

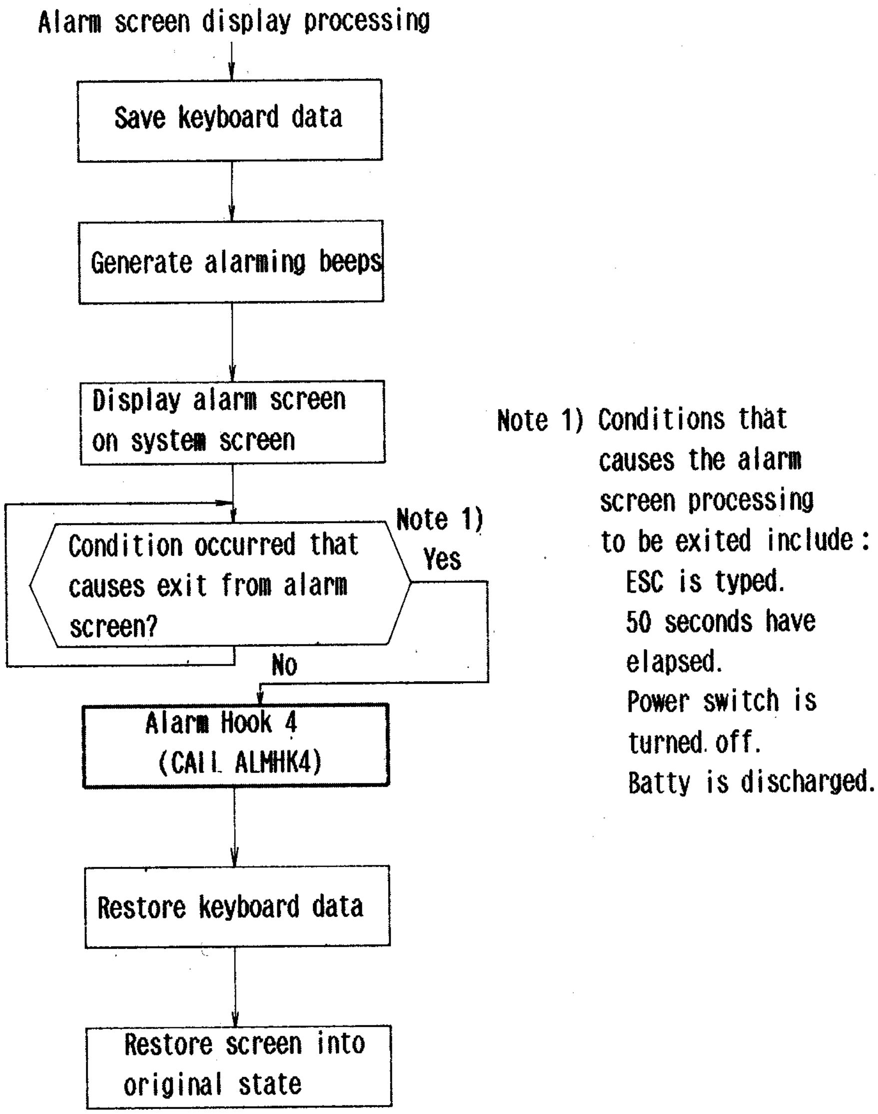


Fig. 4.3.5 Alarm Screen Display Processing

4.3.4 Interrupt Hooks

The PINE has four hooks for interrupt processing. The locations of the hooks are listed below.

HK8251: Within the ART interrupt processing module and immediately before data is received.
ICFHOOK: Within the ICF interrupt processing module
OVFHOOK: Within the OVF interrupt processing module and immediately before blink processing.
EXTHOOK: Within the EXT interrupt processing module

HK8251 is not supported by OS Export version 1.Ø.

4.3.4.1 HK8251

HK8251 is located on OS ROM and provides extended ART interrupt processing. Figure 4.3.6 shows the relationship of HK8251 to interrupt processing.

As seen from the figure, HK8251 just returns control doing nothing when an interrupt occurs unless the RSIOX OPEN function has been executed.

HK8251 is not supported by OS version 1.Ø.

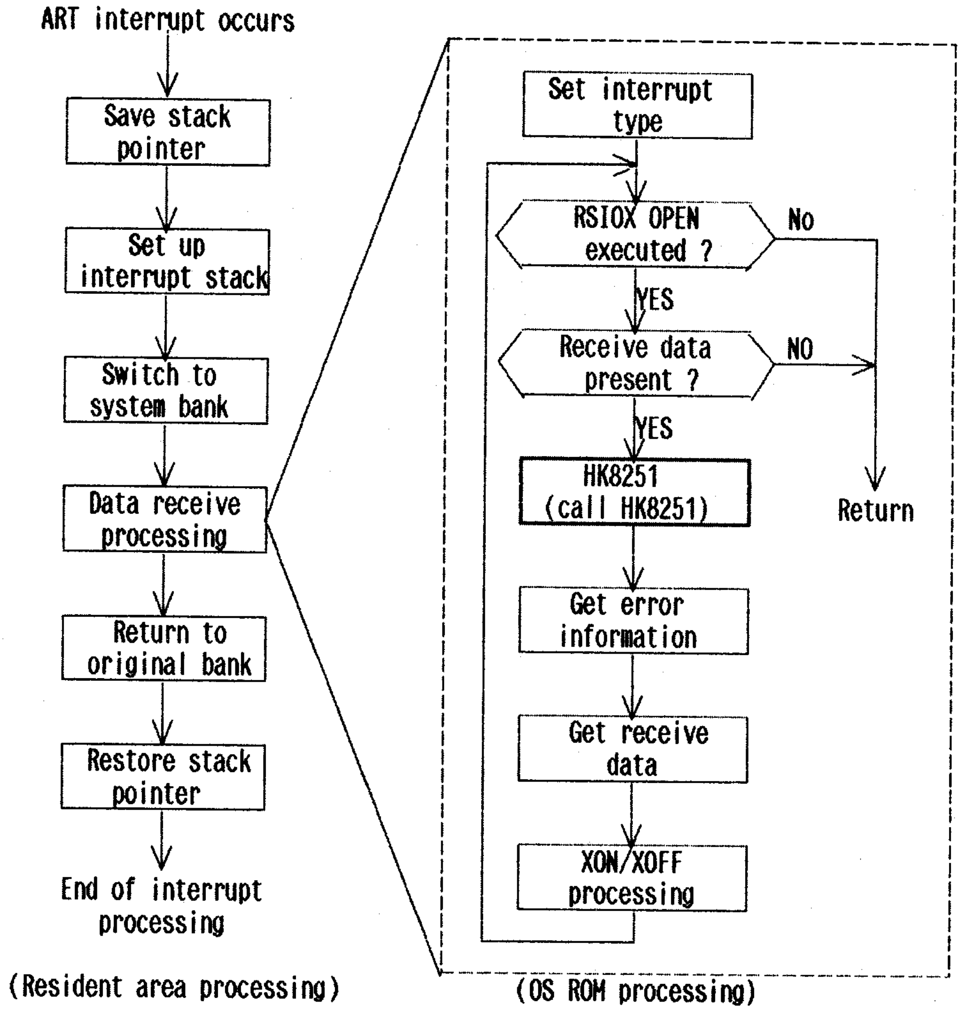


Fig. 4.3.6 ART Interrupt Processing

(2) ICFHOOK, OVFHOOK, and EXTHOOK
During ICF, OVF, and EXT interrupt processing, the active bank is switched to bank Ø (RAM) before the pertinent hook is called. After this, the bank is returned to the original bank.

Figure 4.3.7 shows the relationship of ICFHOOk, OVFHOOk, and EXTHOOK to the interrupt processing modules.

Since the minimum required size of stack area is reserved during ICF, OVF, and EXT processing, it will be necessary to reserve a larger stack area when interrupt processing is to be expanded. Since neither ICF nor EXT interrupt processing modules reset the interrupt signal, it must be reset by the corresponding hook.

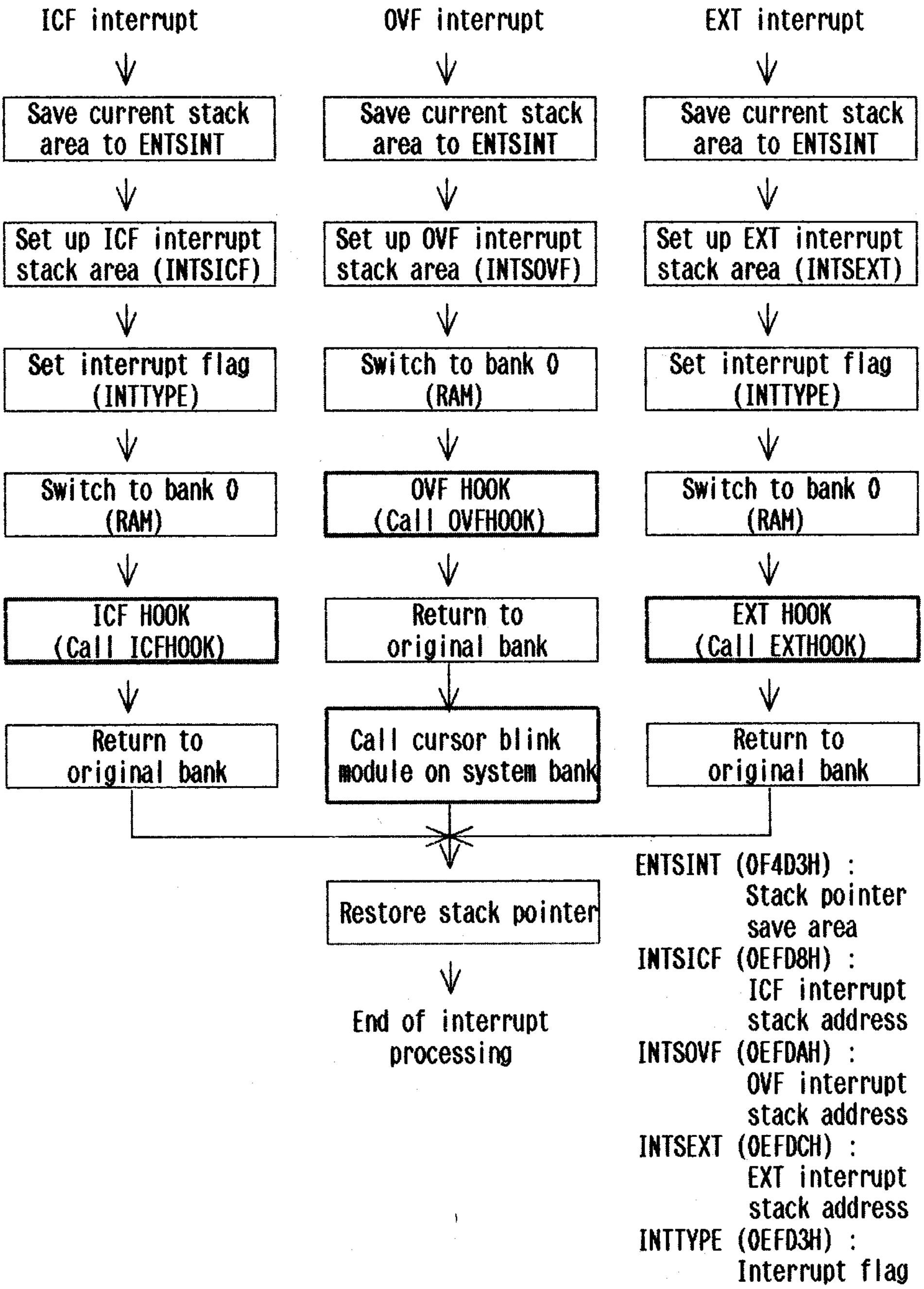


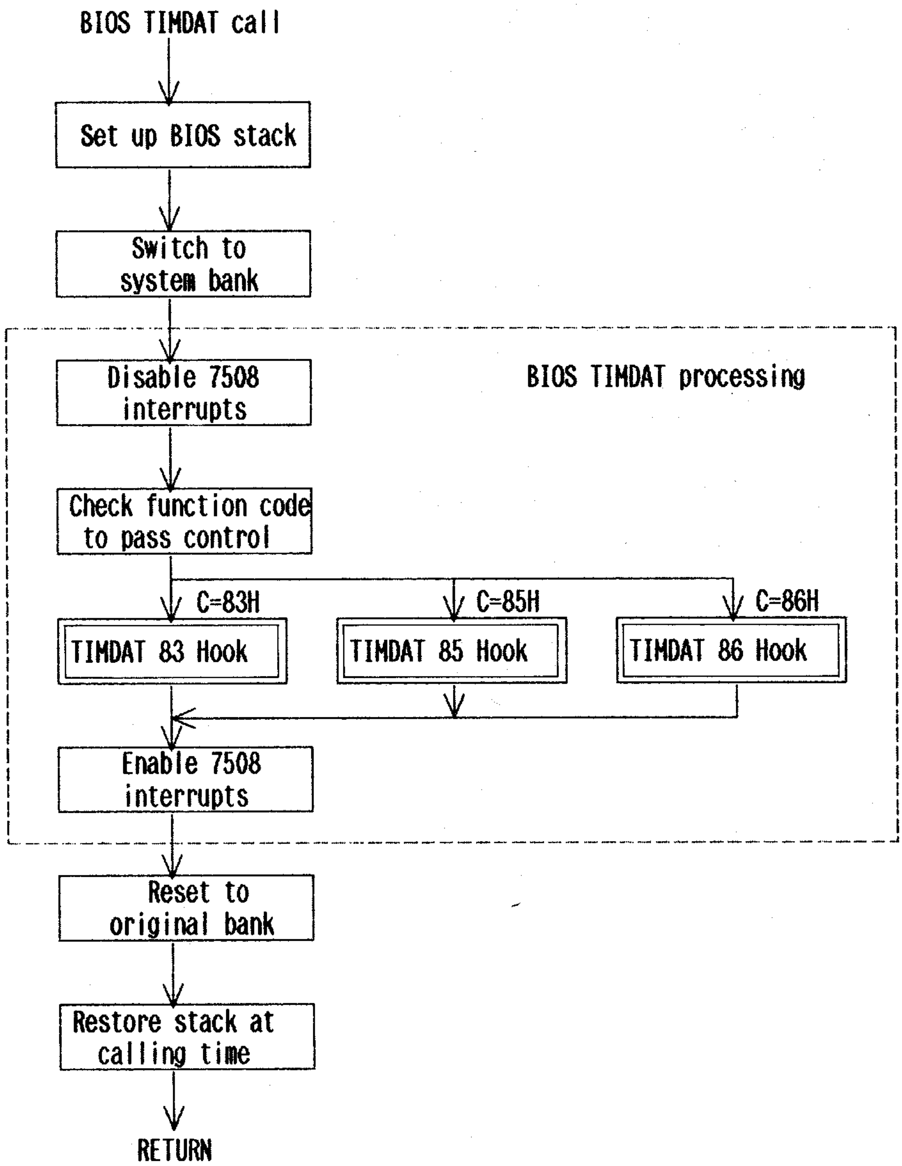
Fig. 4.3.7 ICF/OVF/EXT Interrupt Processing

4.3.5 TIMDAT Hooks

The PINE has the following three hooks for extended BIOS TIMDAT functions.:

TMDT83: Called during BIOS TIMDAT function with C set to 83H.
TMDT85: Called during BIOS TIMDAT function with C set to 85H.
TMDT86: Called during BIOS TIMDAT function with C set to 86H.

These hooks are used to extend the TIMDAT functions. Only the DE registers are preserved when TIMDAT is called.



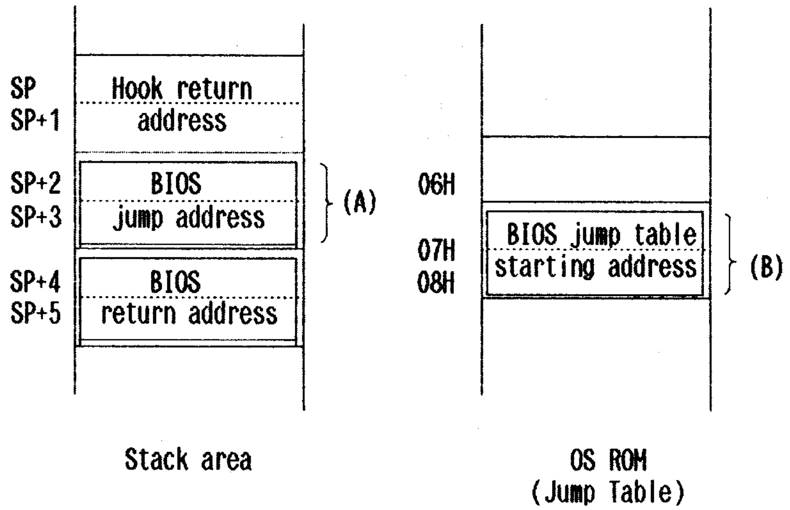
(TIMDAT processing proper is indicated by broken lines.)

Fig. 4.3.8 TIMDAT Processing

4.3.6 BIOS Hook

BIOSHK is taken when BIOS is called by BDOS or by an application program and provides extended BIOS functions.

BIOSHK processing must proceed as follows:
(1) Calculate the BIOS function number.



The logical BIOS number can be calculated using the equation:
{(A) - (B)}/3
where (A) is the BIOS jump address and (B) is the BIOS jump table starting address. For example, the result of the above computation for WBOOT is Ø1H and that for CONST is Ø2H.

(2) Standard BIOS function processing
If the result of step (1) is found to be a standard BIOS call, BIOSHK immediately returns control after restoring the registers into the state before BIOSHK was entered.
(3) Extended BIOS function processing
If the result of step (1) is found to be an extended BIOS call, BIOSHK transfers control to the application program for the extended service. When the system BIOS service is to be executed after the extended processing is completed, BIOSHK restores the registers into the state before BIOSHK was entered and returns control to the calling module. When bypassing the system BIOS service after the extended BIOS processing, BIOSHK returns control directly to the point at the BIOS return address that is stored in the stack area shown in paragraph (1). This allows the system to return control to the program that called BIOS after executing PSTBIOS (including bank switching and stack swapping).

See 3.3.3, "BIOS Hook" for use examples of BIOS hooks.

```
                         ;       ************************************************
                         ;               HOOK SAMPLE PROGRAM
                         ;       ************************************************
                         ;
                         ;       NOTE :
                         ;               This sample program is testing hook.
                         ;
                         ;       <> assemble condition <>
                         ;
                                 .Z80
                         ;
                         ;       <> loading address <>
                         ;
                                 .PHASE  100H
                         ;
                         ;       <> constant values <>
                         ;
                         ;       BIOS entry
                         ;
EB03                     WBOOT           EQU     0EB03H          ; Warm Boot entry
EB09                     CONIN           EQU     WBOOT   +06H    ; Console in entry
EB0C                     CONOUT          EQU     WBOOT   +09H    ; Console out entry
EB69                     CALLX           EQU     WBOOT   +66H    ; Call extra entry
EB87                     RESIDENT        EQU     WBOOT   +84H    ; Resudent entry
                         ;
                         ;       System area
                         ;
EF94                     TOPRAM          EQU     0EF94H          ; Top of User BIOS
EF2D                     USERBIOS        EQU     0EF2DH          ; Size of User BIOS area
F52E                     DISBNK          EQU     0F52EH          ; Destination bank for CALLX
F53E                     RZIER           EQU     0F53EH          ; Interrupt enable register
                         ;
FFC0                     HOOKTBL         EQU     0FFC0H          ; Hook table top address
                         ;
                         ;          k value
                         ;
00FF                     SYSBANK         EQU     0FFH            ; System bank
0000                     BANK0           EQU     000H            ; Bank 0 (RAM)
0001                     BANK1           EQU     001H            ; Bank 1 (ROM capsel 1)
0002                     BANK2           EQU     002H            ; Bank 2 (ROM capsel 2)
                         ;
                         ;
                         ;
CBF0                     UB_HEAD         EQU     0CBF0H          ; Top addr of User BIOS area's header
CBFB                     UB_OVWRITE      EQU     UB_HEAD +11     ;  Over write flag
CBFC                     UB_RELEASE      EQU     UB_HEAD +12     ;  Release address
                         ;
                         ;       OS ROM jump table
                         ;
0006                     BIOSJT          EQU     00006H
003C                     XUSRSCRN        EQU     0003CH
                         ;
                         ;       Resident jump table
                         ;
FF9C                     SELBNK          EQU     0FF9CH
                         ;
000D                     CR              EQU     0DH
000A                     LF              EQU     0AH
0012                     CLS             EQU     12H
                         ;
                         ;       ************************************************
                         ;               MAIN PROGRAM
                         ;       ************************************************
                         ;
                         ;       NOTE :
                         ;               Using system hook jump program.
                         ;
                         ;       CAUTION :
                         ;               This program uses User BIOS area.
                         ;               Usually, you must check that another
                         ;               already used this area. But this program
                         ;               doesn't check it.
                         ;
                         ;               This program doesn't do resetting hook.
                         ;               So, you wish to stop test hook, you must
                         ;               push reset bottom.
                         ;
0100                     MAIN:
0100     C3 03AD                 JP      START           ; Programm start
                         ;
                         ;       *****   Output message data
                         ;
0103                     MSGX:
0103     2A 2A 2A 20             DB      '*** User BIOS size error ***',CR,LF
0107     55 73 65 72
010B     20 42 49 4F
010F     53 20 73 69
0113     7A 65 20 65
0117     72 72 6F 72
011B     20 2A 2A 2A
011F     0D 0A
0121     20 20 53 65             DB      '  Set User BIOS size to 2 K bytes!!'
0125     74 20 55 73
0129     65 72 20 42
012D     49 4F 53 20
0131     73 69 7A 65
0135     20 74 6F 20
0139     32 20 6B 20
013D     62 79 74 65
0141     73 21 21
0144     0D 0A                   DB      CR,LF
0146     00                      DB      00H
                         ;
0147                     MSG0:
0147     12 00                   DB      CLS,00H
                         ;
```

```
0149                                MSG1:
0149     0D 0A                              DB     CR,LF
014B     2A 2A 2A 20                        DB     '*** Hook test program ***'
014F     48 6F 6F 6B
0153     20 74 65 73
0157     74 20 70 72
015B     6F 67 72 61
015F     6D 20 2A 2A
0163     2A
0164     0D 0A                              DB     CR,LF
0166     20 20 20 31                        DB     '   1   -- ALMHKX  test',CR,LF
016A     20 20 20 2D
016E     2D 20 41 4C
0172     4D 48 4B 58
0176     20 20 74 65
017A     73 74 0D 0A
017E     20 20 20 32                        DB     '   2,3 -- Reserve     ',CR,LF
0182     2C 33 20 2D
0186     2D 20 52 65
018A     73 65 72 76
018E     65 20 20 20
0192     20 20 0D 0A
0196     20 20 20 33                        DB     '   3   -- HK8251  test',CR,LF
019A     20 20 20 2D
019E     2D 20 48 4B
01A2     38 32 35 31
01A6     20 20 74 65
01AA     73 74 0D 0A
01AE     20 20 20 34                        DB     '   4   -- ICFHOOK test',CR,LF
01B2     20 20 20 2D
01B6     2D 20 49 43
01BA     46 48 4F 4F
01BE     4B 20 74 65
01C2     73 74 0D 0A
01C6     20 20 20 35                        DB     '   5   -- OVFHOOK test',CR,LF
01CA     20 20 20 2D
01CE     2D 20 4F 56
01D2     46 48 4F 4F
01D6     4B 20 74 65
01DA     73 74 0D 0A
01DE     20 20 20 36                        DB     '   6   -- EXTHOOK test',CR,LF
01E2     20 20 20 2D
01E6     2D 20 45 58
01EA     54 48 4F 4F
01EE     4B 20 74 65
01F2     73 74 0D 0A
01F6     20 20 20 37                        DB     '   7   -- TIMDAT  test',CR,LF
01FA     20 20 20 2D
01FE     2D 20 54 49
0202     4D 44 41 54
0206     20 20 74 65
020A     73 74 0D 0A
020E     20 53 65 6C                        DB     ' Select one of 1 to 7 -- '
0212     65 63 74 20
0216     6F 6E 65 20
021A     6F 66 20 31
021E     20 74 6F 20
0222     37 20 2D 2D
0226     20
0227     00                                 DB     00H
                                    ;
0228                                MSG3:
0228     0D 0A                              DB     CR,LF
022A     0D 0A                              DB     CR,LF
022C     2A 2A 2A 20                        DB     '***  Alarm hook test  ***'
0230     20 41 6C 61
0234     72 6D 20 68
0238     6F 6F 6B 20
023C     74 65 73 74
0240     20 20 2A 2A
0244     2A
0245     0D 0A                              DB     CR,LF
0247     20 20 20 31                        DB     '   1 -- ALMHK1',CR,LF
024B     20 2D 2D 20
024F     41 4C 4D 48
0253     4B 31 0D 0A
0257     20 20 20 32                        DB     '   2 -- ALMHK2',CR,LF
025B     20 2D 2D 20
025F     41 4C 4D 48
0263     4B 32 0D 0A
0267     20 20 20 33                        DB     '   3 -- ALMHK3',CR,LF
026B     20 2D 2D 20
026F     41 4C 4D 48
0273     4B 33 0D 0A
0277     20 20 20 34                        DB     '   4 -- ALMHK4',CR,LF
027B     20 2D 2D 20
027F     41 4C 4D 48
0283     4B 34 0D 0A
0287     20 20 20 35                        DB     '   5 -- ALMHK5',CR,LF
028B     20 2D 2D 20
028F     41 4C 4D 48
0293     4B 35 0D 0A
0297     20 53 65 6C                        DB     ' Select one of 1 to 5 -- '
029B     65 63 74 20
029F     6F 6E 65 20
02A3     6F 66 20 31
02A7     20 74 6F 20
02AB     35 20 2D 2D
02AF     20
02B0     00                                 DB     00H
                                    ;
02B1                                MSG4:
02B1     0D 0A                              DB     CR,LF
02B3     0D 0A                              DB     CR,LF
02B5     2A 2A 2A 20                        DB     '***  Interrupt hook test  ***'
02B9     20 49 6E 74
02BD     65 72 72 75
```

```
02C1    70 74 20 68
02C5    6F 6F 6B 20
02C9    74 65 73 74
02CD    20 20 2A 2A
02D1    2A
02D2    0D 0A                       DB      CR,LF
02D4    20 57 68 65                 DB      ' When interrupt occur,',CR,LF
02D8    6E 20 69 6E
02DC    74 65 72 72
02E0    75 70 74 20
02E4    6F 63 63 75
02E8    72 2C 0D 0A
02EC    20 20 70 72                 DB      '  print message "interrupt".',CR,LF
02F0    69 6E 74 20
02F4    6D 65 73 73
02F8    61 67 65 20
02FC    22 69 6E 74
0300    65 72 72 75
0304    70 74 22 2E
0308    0D 0A
030A    00                          DB      00H
                        ;
030B                    MSG5:
030B    0D 0A                       DB      CR,LF
030D    0D 0A                       DB      CR,LF
030F    2A 2A 2A 20                 DB      '***  TIMDAT test  ***'
0313    20 54 49 4D
0317    44 41 54 20
031B    74 65 73 74
031F    20 20 2A 2A
0323    2A
0324    0D 0A                       DB      CR,LF
0326    20 57 68 65                 DB      ' When timdat routine call,',CR,LF
032A    6E 20 74 69
032E    6D 64 61 74
0332    20 72 6F 75
0336    74 69 6E 65
033A    20 63 61 6C
033E    6C 2C 0D 0A
0342    20 20 70 72                 DB      '  print message "TIMDAT".',CR,LF
0346    69 6E 74 20
034A    6D 65 73 73
034E    61 67 65 20
0352    22 54 49 4D
0356    44 41 54 22
035A    2E 0D 0A
035D    20 20 20 31                 DB      '   1 -- TIMDAT83',CR,LF
0361    20 2D 2D 20
0365    54 49 4D 44
0369    41 54 38 33
036D    0D 0A
036F    20 20 20 32                 DB      '   2 -- TIMDAT85',CR,LF
0373    20 2D 2D 20
0377    54 49 4D 44
037B    41 54 38 35
037F    0D 0A
0381    20 20 20 33                 DB      '   3 -- TIMDAT86',CR,LF
0385    20 2D 2D 20
0389    54 49 4D 44
038D    41 54 38 36
0391    0D 0A
0393    20 53 65 6C                 DB      ' Select one of 1 to 3 -- '
0397    65 63 74 20
039B    6F 6E 65 20
039F    6F 66 20 31
03A3    20 74 6F 20
03A7    33 20 2D 2D
03AB    20
03AC    00                          DB      00H
                        ;
03AD                    START:
03AD    3A EF2D                     LD      A,(USERBIOS)
03B0    FE 08                       CP      08H             ; If User BIOS is smaller than 8 k,
03B2    DA 056C                     JP      C,SZERROR       ;  then error
                        ;
03B5    21 0448                     LD      HL,HOOK         ; Copy programm data for hook
03B8    ED 5B EF94                  LD      DE,(TOPRAM)     ;  from hook to topram (userbios)
03BC    01 0124                     LD      BC,HOOKBTM-HOOK
03BF    F3                          DI                      ; Interrupt disable
03C0    ED B0                       LDIR
03C2    FB                          EI
                        ;
03C3    21 0147                     LD      HL,MSG0         ; Clear screen
03C6    CD 0581                     CALL    DSPMSG
                        ;
03C9                    PTOP1:
03C9    31 2000                     LD      SP,2000H        ; Set stack pointer
                        ;
03CC    0E 01                       LD      C,01H           ; Set resident flag
03CE    CD EB87                     CALL    RESIDENT        ;
                        ;
03D1    21 0149                     LD      HL,MSG1         ; Message (alarm, interrupt, timdat)
03D4    CD 0581                     CALL    DSPMSG
                        ;
03D7                    SELECT2:
03D7    CD 0575                     CALL    CONINF
03DA    CA 058D                     JP      Z,STOPEND       ; If stop, then end
03DD    FE 31                       CP      '1'
03DF    38 F6                       JR      C,SELECT2       ; If select error, then retry
03E1    28 08                       JR      Z,ALARM         ; Alarm select
03E3    FE 36                       CP      '6'
03E5    28 2A                       JR      Z,TIMDAT        ; TIMDAT select
03E7    30 EE                       JR      NC,SELECT2      ; If select error, then retry
03E9    18 18                       JR      INTERRUPT       ; else interrupt
                        ;
                        ;           *****  Alarm hook test *****
                        ;
```

```
03EB                          ALARM:
03EB   21 0228                        LD     HL,MSG3         ; Message (alarm hook 1 -- 5)
03EE   CD 0581                        CALL   DSPMSG
                              ;
03F1                          SELECT3:
03F1   CD 0575                        CALL   CONINF
03F4   CA 058D                        JP     Z,STOPEND       ; If stop, then end
03F7   FE 31                          CP     '1'
03F9   38 F6                          JR     C,SELECT3       ; If select error, then retry
03FB   FE 36                          CP     '6'
03FD   30 F2                          JR     NC,SELECT3      ; If select error, then retry
                              ;
03FF   D6 31                          SUB    '1'             ; A reg. is 0..4
0401   18 28                          JR     SETHOOK         ; Goto hook setting process
                              ;
                              ;       *****  Interrupt hook test      *****
                              ;
0403                          INTERRUPT:
0403   F5                             PUSH   AF              ; Save selecting number
0404   21 02B1                        LD     HL,MSG4
0407   CD 0581                        CALL   DSPMSG          ; Interrupt hook select message
040A   F1                             POP    AF
040B   D6 32                          SUB    '2'
040D   C6 05                          ADD    A,05H           ; A reg. is 5..9
040F   18 1A                          JR     SETHOOK         ; Goto hook setting process
                              ;
                              ;       ***** TIMDAT hook test  *****
                              ;
0411                          TIMDAT:
0411   21 030B                        LD     HL,MSG5
0414   CD 0581                        CALL   DSPMSG          ; Select TIMDAT message
                              ;
0417                          SELECT4:
0417   CD 0575                        CALL   CONINF
041A   CA 058D                        JP     Z,STOPEND       ; If stop, then end
                              ;
041D   FE 31                          CP     '1'
041F   38 F6                          JR     C,SELECT4       ; If select error, then retry
0421   FE 34                          CP     '4'
0423   30 F2                          JR     NC,SELECT4      ; If select error, then retry
                              ;
0425   D6 31                          SUB    '1'
0427   C6 0A                          ADD    A,10            ; A reg. is 10..12
0429   18 00                          JR     SETHOOK         ; Goto hook setting process
                              ;
                              ;       *****  Set hook data for each select  ****
                              ;              A -- Hook logical number (0 -- 12)
042B                          SETHOOK:
042B   4F                             LD     C,A             ; Get hook data addr
042C   87                             ADD    A,A
042D   57                             LD     D,A
042E   81                             ADD    A,C
042F   4F                             LD     C,A
0430   06 00                          LD     B,00H
0432   21 FFC0                        LD     HL,HOOKTBL
0435   09                             ADD    HL,BC           ; HOOKTBL + A*3 --> HL
0436   E5                             PUSH   HL              ; Push target hook table
                              ;
0437   2A EF94                        LD     HL,(TOPRAM)
043A   87                             ADD    A,A
043B   4F                             LD     C,A
043C   09                             ADD    HL,BC           ; (TOPRAM) + A*6 --> HL
043D   EB                             EX     DE,HL
                              ;
043E   E1                             POP    HL              ; Set hook jump addr
043F   F3                             DI                     ; Interrupt disable
0440   23                             INC    HL
0441   73                             LD     (HL),E
0442   23                             INC    HL
0443   72                             LD     (HL),D
0444   FB                             EI
                              ;
0445   C3 03C9                        JP     PTOP1
                              ;
                              ;       *****  User BIOS data  *****
                              ;              This part is copyed into user bios area
0448                          HOOK:
0448   E5                             PUSH   HL
0449   21 C4A4                        LD     HL,0C400h+PDATA1-HOOK
044C   18 48                          JR     HOOKSTART
                              ;
044E   E5                             PUSH   HL
044F   21 C4AD                        LD     HL,0C400H+PDATA2-HOOK
0452   18 42                          JR     HOOKSTART
                              ;
0454   E5                             PUSH   HL
0455   21 C4B6                        LD     HL,0C400H+PDATA3-HOOK
0458   18 3C                          JR     HOOKSTART
                              ;
045A   E5                             PUSH   HL
045B   21 C4BF                        LD     HL,0C400H+PDATA4-HOOK
045E   18 36                          JR     HOOKSTART
                              ;
0460   E5                             PUSH   HL
0461   21 C4C8                        LD     HL,0C400H+PDATA5-HOOK
0464   18 30                          JR     HOOKSTART
                              ;
0466   E5                             PUSH   HL
0467   21 C4D1                        LD     HL,0C400H+PDATA6-HOOK
046A   18 2A                          JR     HOOKSTART
                              ;
046C   E5                             PUSH   HL
046D   21 C4DA                        LD     HL,0C400H+PDATA7-HOOK
0470   18 24                          JR     HOOKSTART
                              ;
0472   E5                             PUSH   HL
0473   21 C4E3                        LD     HL,0C400H+PDATA8-HOOK
```

```
0476        18 1E                       JR      HOOKSTART
                             ;
0478        E5                          PUSH    HL
0479        21 C4ED                     LD      HL,0C400H+PDATA9-HOOK
047C        18 18                       JR      HOOKSTART
                             ;
047E        E5                          PUSH    HL
047F        21 C4F7                     LD      HL,0C400H+PDATA10-HOOK
0482        18 12                       JR      HOOKSTART
                             ;
0484        E5                          PUSH    HL
0485        21 C501                     LD      HL,0C400H+PDATA11-HOOK
0488        18 0C                       JR      HOOKSTART
                             ;
048A        E5                          PUSH    HL
048B        21 C50C                     LD      HL,0C400H+PDATA12-HOOK
048E        18 06                       JR      HOOKSTART
                             ;
0490        E5                          PUSH    HL
0491        21 C517                     LD      HL,0C400H+PDATA13-HOOK
0494        18 00                       JR      HOOKSTART
                             ;
                             ;
0496                         HOOKSTART:
0496        ED 73 C522                  LD      (0C400H+SAVESP-HOOK),SP
                                                                ; Set stack pointer
                             ;
049A        31 CAEF                     LD      SP,0CAEFH
                             ;
049D        E5                          PUSH    HL              ; Save all registers
049E        D5                          PUSH    DE
049F        C5                          PUSH    BC
04A0        F5                          PUSH    AF
                             ;
04A1        3A F53E                     LD      A,(RZIER)       ; Interrupt disable
04A4        F5                          PUSH    AF
04A5        AF                          XOR     A
04A6        D3 04                       OUT     (4),A
                             ;
04A8        0E FF                       LD      C,0FFH
04AA        CD FF9C                     CALL    SELBNK          ; Select OS bank
04AD        C5                          PUSH    BC              ; Save old bank information
                             ;
                             ;                                  ; Print message
04AE        E5                          PUSH    HL
04AF        CD 003C                     CALL    XUSRSCRN        ; Change to user screen
04B2        E1                          POP     HL
                             ;
04B3        CD C482                     CALL    0C400h+XCONOUT-HOOK
                             ;
04B6        C1                          POP     BC
04B7        CD FF9C                     CALL    SELBNK          ; Recover old bank
                             ;
04BA        F1                          POP     AF              ; Recover interrupt register
04BB        32 F53E                     LD      (RZIER),A
04BE        D3 04                       OUT     (4),A
                             ;
04C0        F1                          POP     AF              ; Restore all registers
04C1        C1                          POP     BC
04C2        D1                          POP     DE
04C3        E1                          POP     HL
                             ;
04C4        ED 7B C522                  LD      SP,(0C400H+SAVESP-HOOK)
                                                                ; Restore stack pointer
                             ;
04C8        E1                          POP     HL
04C9        C9                          RET
                             ;
04CA                         XCONOUT:
04CA        1E 0C                       LD      E,0CH           ; Conout function
04CC        18 02                       JR      XPRINT0
                             ;
04CE                         XPRINT:
04CE        1E 0F                       LD      E,0FH           ; List function
                             ;
04D0                         XPRINT0:
04D0        4E                          LD      C,(HL)          ; DATA --> C
04D1        79                          LD      A,C
04D2        B7                          OR      A
04D3        C8                          RET     Z               ; Data end?
                             ;
04D4        CD C492                     CALL    0C400H+ROMBIOS-HOOK
                                                                ; Call ROM BIOS
04D7        23                          INC     HL              ; Pointer update
04D8        18 F6                       JR      XPRINT0
                             ;
                             ;          *****   Select target rom bios   *****
                             ;                  E -- Function number
                             ;
04DA                         ROMBIOS:
04DA        E5                          PUSH    HL              ; Save all registers
04DB        D5                          PUSH    DE
04DC        C5                          PUSH    BC
                             ;
04DD        2A 0007                     LD      HL,(BIOSJT+1)   ; Get ROM BIOS jump addr
04E0        16 00                       LD      D,00H
04E2        19                          ADD     HL,DE
04E3        11 C4A0                     LD      DE,0C400H+RETADDR-HOOK
04E6        D5                          PUSH    DE              ; Push return addr
04E7        E9                          JP      (HL)            ; Go !!
                             ;
04E8                         RETADDR:
04E8        C1                          POP     BC              ; Restore all registers
04E9        D1                          POP     DE
04EA        E1                          POP     HL
04EB        C9                          RET
                             ;
                             ;
04EC                         PDATA1:
04EC        41 4C 41 52                 DB      'ALARM1',CR,LF,0
```

```
04F0    4D 31 0D 0A
04F4    00
04F5                            PDATA2:
04F5    41 4C 41 52                     DB      'ALARM2',CR,LF,0
04F9    4D 32 0D 0A
04FD    00
04FE                            PDATA3:
04FE    41 4C 41 52                     DB      'ALARM3',CR,LF,0
0502    4D 33 0D 0A
0506    00
0507                            PDATA4:
0507    41 4C 41 52                     DB      'ALARM4',CR,LF,0
050B    4D 34 0D 0A
050F    00
0510                            PDATA5:
0510    41 4C 41 52                     DB      'ALARM5',CR,LF,0
0514    4D 35 0D 0A
0518    00
0519                            PDATA6:
0519    54 4D 48 4F                     DB      'TMHOOK',CR,LF,0
051D    4F 4B 0D 0A
0521    00
0522                            PDATA7:
0522    48 4B 38 32                     DB      'HK8251',0ah,0dh,0
0526    35 31 0A 0D
052A    00
052B                            PDATA8:
052B    49 43 46 48                     DB      'ICFHOOK',0ah,0dh,0
052F    4F 4F 4B 0A
0533    0D 00
0535                            PDATA9:
0535    4F 56 46 48                     DB      'OVFHOOK',0ah,0dh,0
0539    4F 4F 4B 0A
053D    0D 00
053F                            PDATA10:
053F    45 58 54 48                     DB      'EXTHOOK',CR,LF,0
0543    4F 4F 4B 0D
0547    0A 00
0549                            PDATA11:
0549    54 49 4D 44                     DB      'TIMDAT83',CR,LF,0
054D    41 54 38 33
0551    0D 0A 00
0554                            PDATA12;
0554    54 49 4D 44                     DB      'TIMDAT85',CR,LF,0
0558    41 54 38 35
055C    0D 0A 00
055F                            PDATA13:
055F    54 49 4D 44                     DB      'TIMDAT86',CR,LF,0
0563    41 54 38 36
0567    0D 0A 00
                                ;
056A                            SAVESP:
056A    0000                            DW      0000H
                                ;
056C                            HOOKBTM:
                                ;
                                ;
                                ;
                                ;
056C                            SZERROR:
056C    21 0103                         LD      HL,MSGX         ; Error message display
056F    CD 0581                         CALL    DSPMSG
0572    C3 058D                         JP      STOPEND
                                ;
                                ;       *****   Console input routine   *****
                                ;
0575                            CONINF:
0575    CD EB09                         CALL    CONIN           ; Console in
0578    FE 03                           CP      03H
057A    F5                              PUSH    AF              ; If stop, then z-flag on
057B    4F                              LD      C,A
057C    CD EB0C                         CALL    CONOUT          ; Display inputing char
057F    F1                              POP     AF
0580    C9                              RET
                                ;
                                ;       *****   Console output routine  *****
                                ;               IN :    HL -- Conout message top addr.
0581                            DSPMSG:
0581    4E                              LD      C,(HL)
0582    79                              LD      A,C
0583    B7                              OR      A               ; If data is 0,
0584    C8                              RET     Z               ;  then end of data.
                                ;
0585    E5                              PUSH    HL
0586    CD EB0C                         CALL    CONOUT          ; Console output
0589    E1                              POP     HL
058A    23                              INC     HL
058B    18 F4                           JR      DSPMSG
                                ;
                                ;       *****   Ending process  *****
                                ;
058D                            STOPEND
058D    0E 00                           LD      C,00H           ; Reset resident flag
058F    CD EB87                         CALL    RESIDENT
0592    C3 EB03                         JP      WBOOT           ;
                                ;
                                        END
```

4.4 Bank Switching

4.4.1 General

The PINE is provided with four banks of memory which are managed by the PINE CP/M operating system. Normal application programs can access the PINE memory without being aware of these memory banks. There are some advanced application programs which need to control the memory banks directly. For such application programs, this section describes the facilities to control the memory banks and shows how to use those facilities.

4.4.2 Bank Structure

4.4.2.1 Bank types

The PINE memory is divided into the following four banks:

(1) System bank: Accommodates the OS ROM and the higher portion of the RAM.
(2) Bank Ø: RAM.
(3) Bank 1: Application program ROM and portion of RAM.
(4) Bank 2: Application program ROM and portion of RAM.

The four memory banks provide 64K bytes of RAM and 96K bytes (maximum) of ROM.

Banks 1 and 2 are available in three configurations depending on the capacity of the ROM area (8K, 16K, and 32K bytes). PINE OS establishes the memory bank configuration at power-on time by reading the capacity of the ROM area on banks 1 and 2 from their header area. The ROM capacity for bank 1 is 32K bytes (for BASIC interpreter) as standard. This ROM area may be replaced by an application ROM. Figure 4.4.1 shows the PINE memory space.

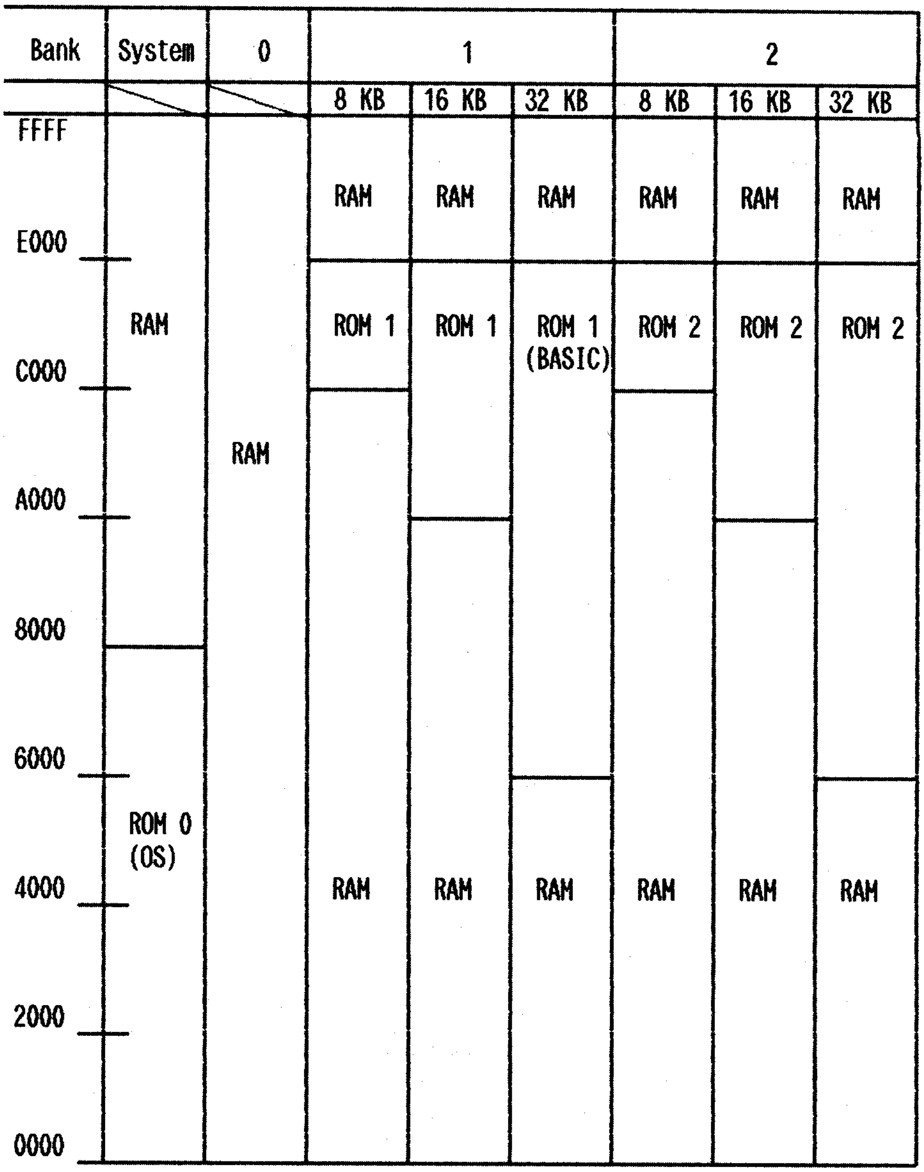
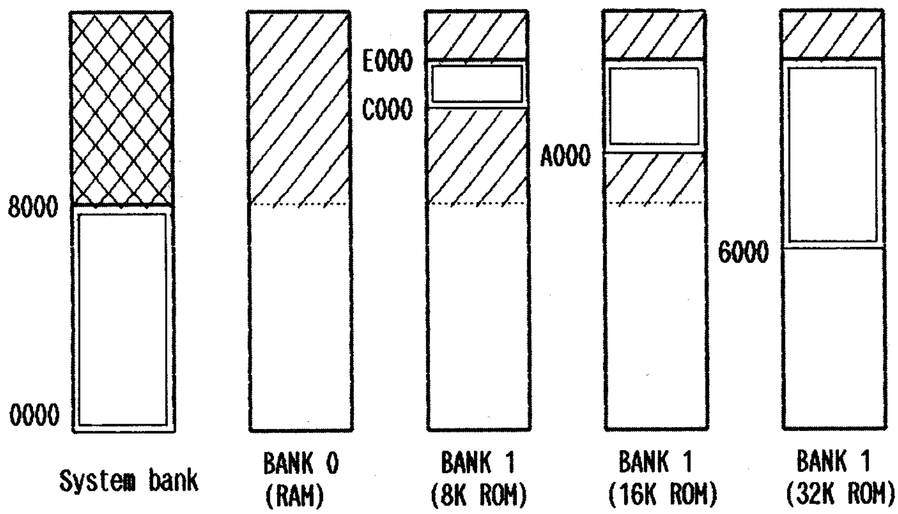


Fig. 4.4.1 PINE Memory Space

4.4.2.2 Common RAM areas

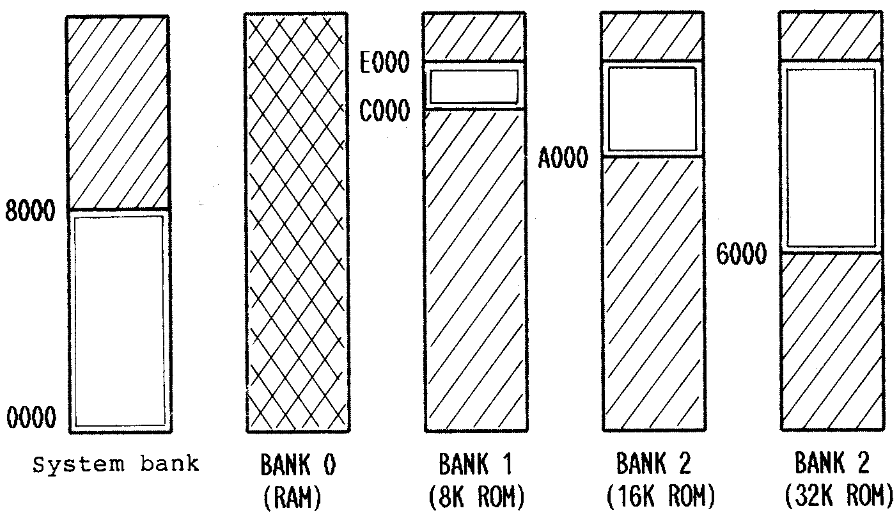
The following areas can be used as common area when banks are switched:

(1) Switching from system bank to bank Ø, 1, or 2



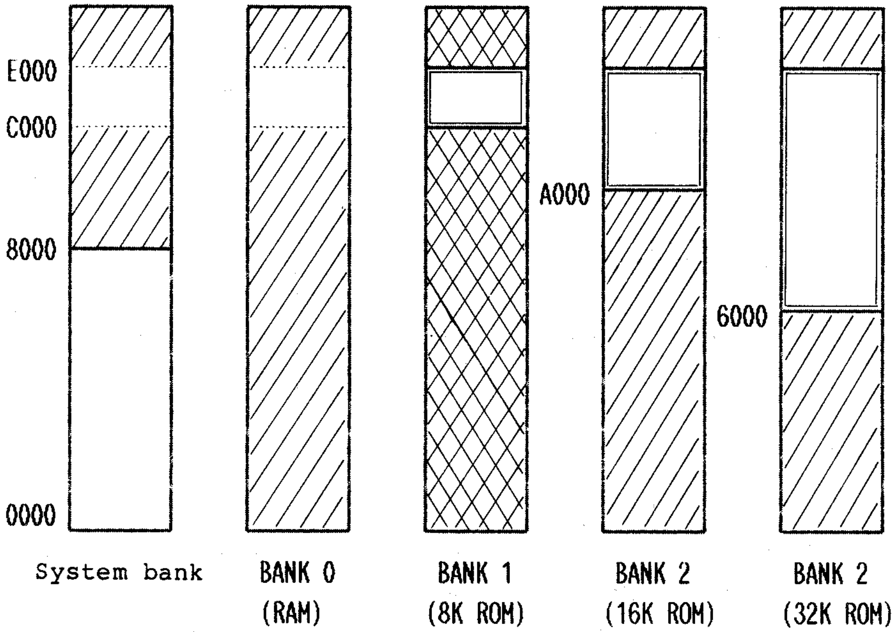
Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. Areas enclosed in double boxes are made up of ROM. Bank 2 has the same configuration as bank 1.

(2) Switching from bank Ø to system bank, bank 1 or 2



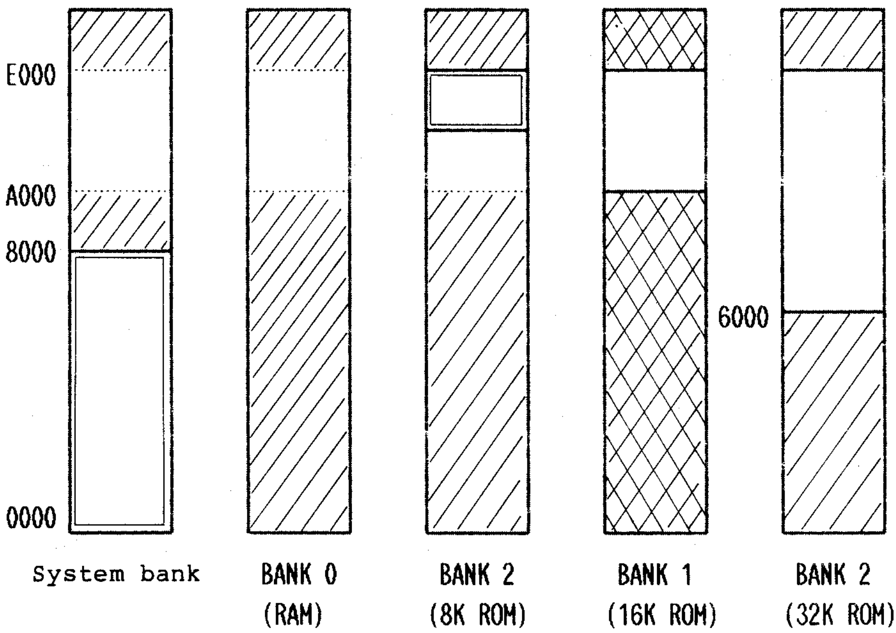
Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. Bank 2 has the same configuration as bank 1.

(3) Switching from bank 1 to system bank, bank Ø or 2 (8K ROM)



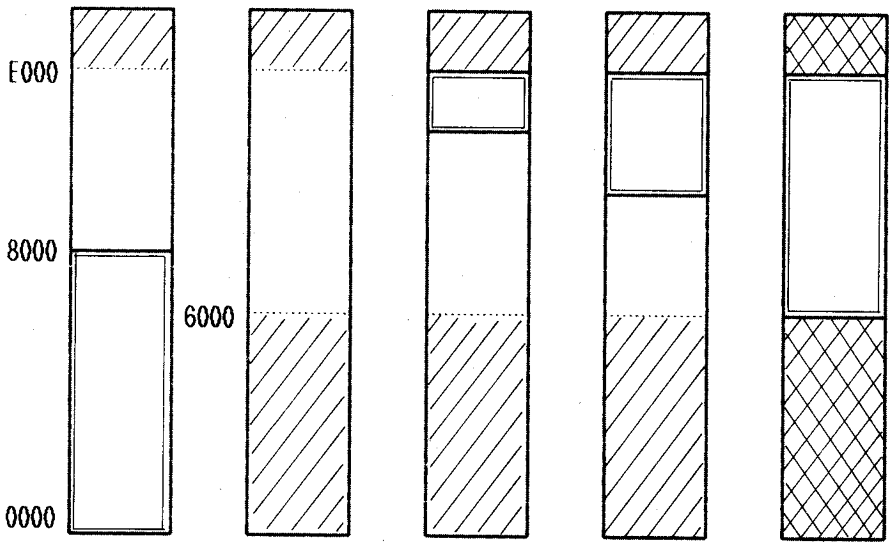
Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. On bank 2 (8K ROM), the area excluding the ROM area can be used as a common area.

(4) Switching from bank 1 to system bank, bank Ø or 2 (16K ROM)



Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. On bank 2 (16K ROM), the area excluding the ROM area can be used as a common area.

(5) Switching from bank 1 to system bank, bank Ø or 2 (32K ROM)



Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. On bank 2 (32K ROM), the area excluding the ROM area can be used as a common area.

4.4.2.3 Programming notes

(1) Banks 1 and 2 have different memory configurations depending on the capacity of the ROM devices in the ROM capsule. PINE OS establishes the memory bank configuration at power-on time by reading the capacity of the ROM area on banks 1 and 2. More specifically, the OS searches for the ROM header and, if one is found, read the ROM size from the header. If no header is found, the OS regards the bank as having no ROM. See the description of ROM capsules in Section 3.8, "Disk Storage" for the ROM header.

(2) The OS switches banks based on the bank information (BNKRGS, BNKRG1, BNKRG2, and BNKRG3) that are examined in paragraph (1).

4.4.3 Utility Routines

PINE OS supplies the following two types of utility routines for bank management:

1. Bank subroutines using BIOS functions
2. Bank subroutines which controls banks directly.

The application program must select one of these subroutine types depending on the purpose of its bank control.

4.4.3.1 Bank switching using BIOS

Bank switching using BIOS is used by ordinary application programs to exchange data between banks. PINE OS provides six BIOS functions for bank switching. Their functions are summarized in the table below.

The application program need not consider the stack area to be used during bank switching because the OS reserves it by itself. See Section 3.4, "BIOS Details" for details of the six BIOS functions.

Name	Address	Description
LOADX	WBOOT + 5AH or ØEB5DH	Reads one byte from the given address on the given bank.
STORX	WBOOT + 5DH or ØEB6ØH	Writes one byte into the given address on the given bank.
LDIRX	WBOOT + 6ØH or ØEB63H	Writes the given number of bytes at the given address on the given bank into the bank Ø area at the given address.
JUMPX	WBOOT + 63H or ØEB66H	Causes the CPU to jump to the given address on the given bank.
CALLX	WBOOT + 66H or ØEB69H	Calls the module at the given address on the given bank.
MEMORY	WBOOT + 4EH or ØEB51H	Gets the current bank information.

4.4.3.2 Bank switching using direct bank control

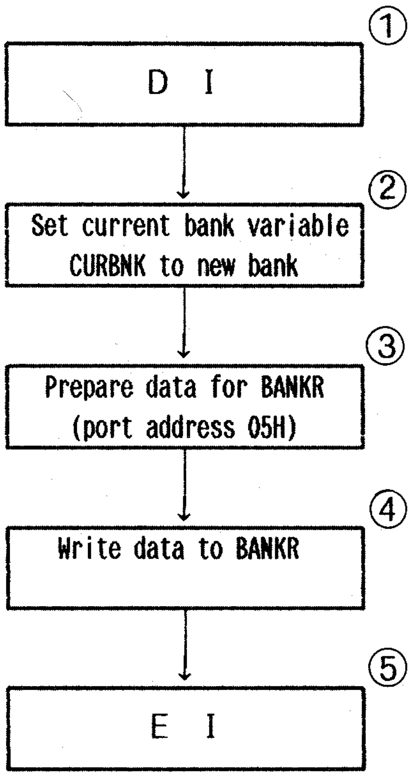
Bank switching using direct bank control is used by special application programs (extending BIOS or interrupt handling functions) which need to exercise direct control over banks. PINE OS provides six bank switching utility routines. Their functions are summarized in the table below.

When using these routines, the application program must reserve a stack area at a location in which it is accessible to the new bank. See Section 4.2, "Jump Tables" for details on the bank switching utility routines.

Name	Address	Description
SELBNK	ØFF9CH	Switches to the given bank.
LDAXX	ØFF9FH	Reads one byte from the given address on the given bank.
STAXX	ØFFA2H	Writes one byte into the given address on the given bank.
LDIRXX	ØFFA5H	Writes the given number of bytes at the given address on the given bank into the bank Ø area at the given address.
JUMPXX	ØFFA8H	Causes the CPU to jump to the given address on the given bank.
CALLXX	ØFFABH	Calls the module at the given address on the given bank.

4.4.3.3 Programming notes

The application program must use the following steps to switch banks using no OS routine:



BANKR (Bank Register) is a write-only register assigned to I/O port address Ø5H. Its format is shown below.