NEC

Application Note

Code Banking

78K0 Series

Document No. U18762EE1V1AN00
Date published May 2007

Printed in Germany

Legal Notes

- **The information in this document is current as of August, 2007. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**

- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.

- NEC Electronics does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from the use of NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.

- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.

- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.

- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".

- The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.
  "Standard": Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots.
  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime

systems, safety equipment and medical equipment (not specifically designed for life support).
"Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

(Note)
(1) "NEC Electronics" as used in this statement means NEC Electronics Corporation and also includes its majority-owned subsidiaries.
(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

Notes for CMOS Devices

1. **VOLTAGE APPLICATION WAVEFORM AT INPUT PIN**
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between VIL (MAX) and VIH (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between VIL (MAX) and VIH (MIN).

2. **HANDLING OF UNUSED INPUT PINS**
   Unconnected CMOS device inputs can result in malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to VDD or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

3. **PRECAUTION AGAINST ESD**
   A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and to quickly dissipate it should it occur. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

4. **STATUS BEFORE INITIALIZATION**
   Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.

5. **POWER ON/OFF SEQUENCE**
   In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current. The correct power on/off sequence must be

judged separately for each device and according to related specifications governing the device.

6. **INPUT OF SIGNAL DURING POWER OFF STATE**
Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact theNEC office in your country to obtain a list of authorized representatives anddistributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and otherlegal issues may also vary from country to country.

NEC Electronics Corporation
1753, Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa 211-8668, Japan
Tel: 044 4355111
http://www.necel.com/

[America]

NEC Electronics America, Inc.
2880 Scott Blvd.
Santa Clara, CA 95050-2554,
U.S.A.
Tel: 408 5886000
http://www.am.necel.com/

[Europe]

NEC Electronics (Europe) GmbH
Arcadiastrasse 10
40472 Düsseldorf, Germany
Tel: 0211 65030
http://www.eu.necel.com/

United Kingdom Branch
Cygnus House, Sunrise Parkway
Linford Wood, Milton Keynes
MK14 6NP, U.K.
Tel: 01908 691133

Succursale Française
9, rue Paul Dautier, B.P. 52
78142 Velizy-Villacoublay Cédex
France
Tel: 01 30675800

Sucursal en España
Juan Esplandiu, 15
28007 Madrid, Spain
Tel: 091 5042787

Tyskland Filial
Täby Centrum
Entrance S (7th floor)
18322 Täby, Sweden
Tel: 08 6387200

Filiale Italiana
Via Fabio Filzi, 25/A
20124 Milano, Italy
Tel: 02 667541

Branch The Netherlands
Steijgerweg 6
5616 HS Eindhoven,
The Netherlands
Tel: 040 2654010

[Asia & Oceania]

NEC Electronics (China) Co., Ltd
7th Floor, Quantum Plaza, No. 27
ZhiChunLu Haidian District,
Beijing 100083, P.R.China
Tel: 010 82351155
http://www.cn.necel.com/

NEC Electronics Shanghai Ltd.
Room 2511-2512, Bank of China
Tower,
200 Yincheng Road Central,
Pudong New Area,
Shanghai 200120, P.R. China
Tel: 021 58885400
http://www.cn.necel.com/

NEC Electronics Hong Kong Ltd.
12/F., Cityplaza 4,
12 Taikoo Wan Road, Hong Kong
Tel: 2886 9318
http://www.hk.necel.com/

NEC Electronics Taiwan Ltd.
7F, No. 363 Fu Shing North Road
Taipei, Taiwan, R.O.C.
Tel: 02 27192377

NEC Electronics Singapore Pte. Ltd.
238A Thomson Road,
#12-08 Novena Square,
Singapore 307684
Tel: 6253 8311
http://www.sg.necel.com/

NEC Electronics Korea Ltd.
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku, Seoul,
135-080, Korea Tel: 02-558-3737
http://www.kr.necel.com/

Table of Contents

Chapter 1 Overview

1.1 Introduction

If an application needs more than 60kB ROM the 78K0 Series reaches the limit of direct addressable memory. To overcome this limit the complete software development environment (Compiler, Linker, In-Circuit Emulator) supports a code expansion technique called banked memory.

Chapter 2 Memory Bank Select Function

For NEC 78K0 Series up to 6 banks can be addressed with a bank size of 16kB. This results to a ROM size of up to 96kB banked memory plus 32kB common memory.

The main task of the compiler and linker is to place different parts of code into the same physical address range. Each time a banked function has to be called, the correct bank has to be selected. This is realized by a special function call procedure.

2.1 Memory Bank

The memory area from 0000H to 7FFFH should be used as basic common code area, where the bank switching is located in. The banked code area is located in the address range 8000H to BFFFH.

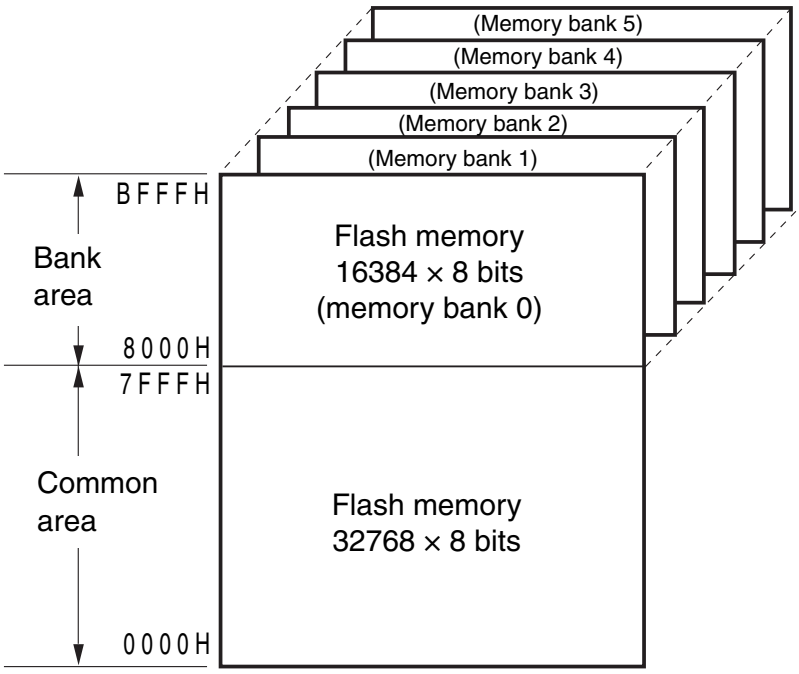
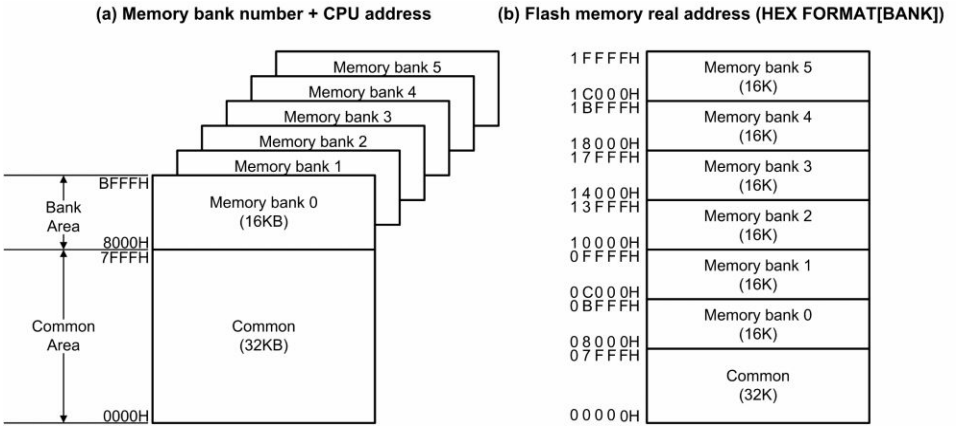


Figure 2-1 Internal Flash Memory Configuration

On 78K0 products which support the code banking, addresses can be viewed in the following two different ways:

- Memory bank number + CPU address
- Flash memory real address (HEX format [BANK])



The “Memory bank number + CPU address” is represented with a gap in the address space, while the flash memory real address is shown without gap in the address space.

The “Memory bank number + CPU address” is used for addressing in the user program.

Note that the HEX file output uses by default the flash memory real address.

For address representation of the other tools such as the simulator and the debugger, pls. refer to the table below:

Table 2-1 Memory Bank Address Representation

Memory Bank Number	CPU Address	Flash Memory Real Address	Address Representation in Debugger
Memory bank 0	08000H-0BFFFH	08000H-0BFFFH	08000H-0BFFFH
Memory bank 1		0C000H-0FFFFH	18000H-1BFFFH
Memory bank 2		10000H-13FFFH	28000H-2BFFFH
Memory bank 3		14000H-17FFFH	38000H-3BFFFH
Memory bank 4		18000H-1BFFFH	48000H-4BFFFH
Memory bank 5		1C000H-1FFFFH	58000H-5BFFFH

2.2 Memory Bank Select Register (BANK)

The memory bank select register (BANK) is used to select a memory bank to be used.

Access BANK can be set by an 8-bit memory manipulation instruction.

Address $FFF3_H$

Initial Value Reset signal generation clears BANK to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	BANK2	BANK1	BANK0
R	R	R	R	R	R/W	R/W	R/W

BANK2	BANK1	BANK0	Description
0	0	0	Common area (32K) + memory bank 0 (16K)
0	0	1	Common area (32K) + memory bank 1 (16K)
0	1	0	Common area (32K) + memory bank 2 (16K)
0	1	1	Common area (32K) + memory bank 3 (16K)
1	0	0	Common area (32K) + memory bank 4 (16K)
1	0	1	Common area (32K) + memory bank 5 (16K)
Other than above			Setting prohibited

Caution Be sure to change the value of the BANK register in the common area (0000H to 7FFFH). If the value of the BANK register is changed in the bank area (8000H to BFFFH), an inadvertent program loop occurs in the CPU. Therefore, never change the value of the BANK register in the bank area.

2.3 Selecting Memory Bank

The memory bank selected by the memory bank select register (BANK) is reflected on the bank area and can be addressed. Therefore, to access a memory bank different from the one currently selected, that memory bank must be selected by using the BANK register.

The value of the BANK register must not be changed in the bank area (8000H to BFFFH). Therefore, to change the memory bank, branch an instruction to the common area (0000H to 7FFFH) and change the value of the BANK register in that area.

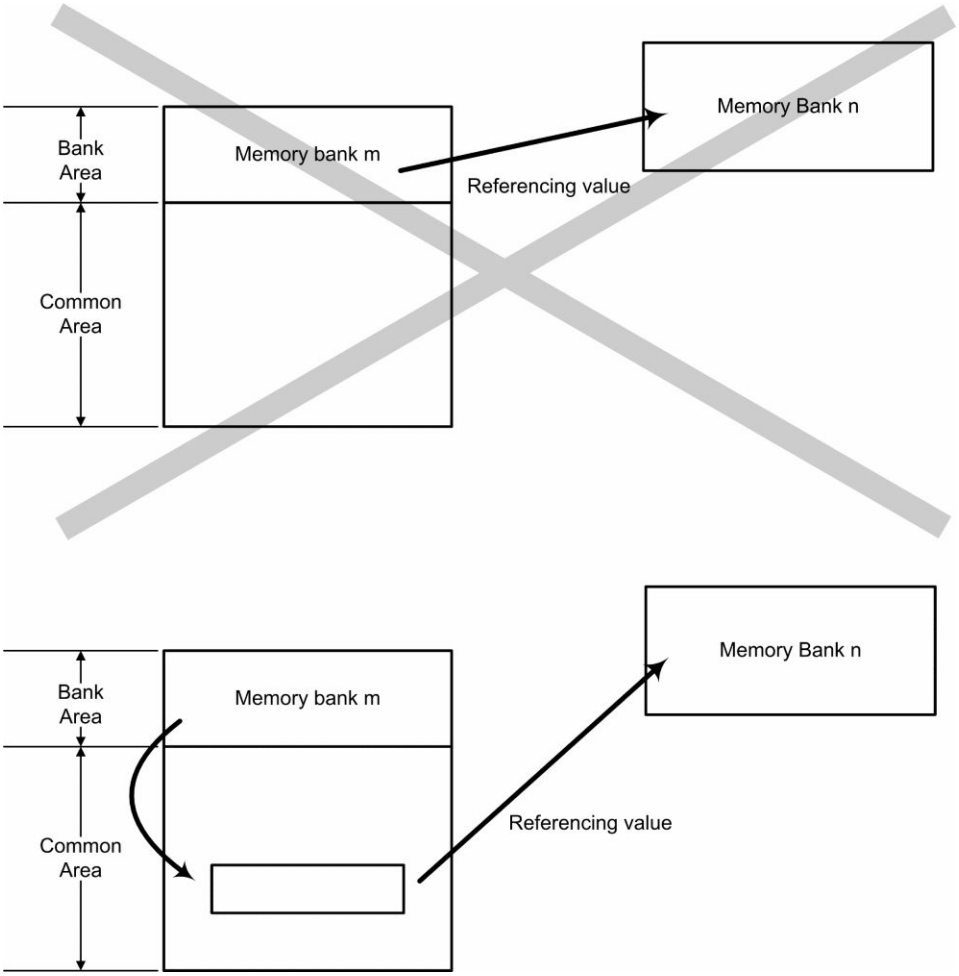
Caution

1. Instructions cannot be fetched between different memory banks
2. Branching and accessing cannot be directly executed between different memory banks. Execute branching or accessing between different memory banks via the common area.
3. Allocate interrupt servicing in the common area.
4. An instruction that extends from 7FFFH to 8000H can only be executed in memory bank 0.

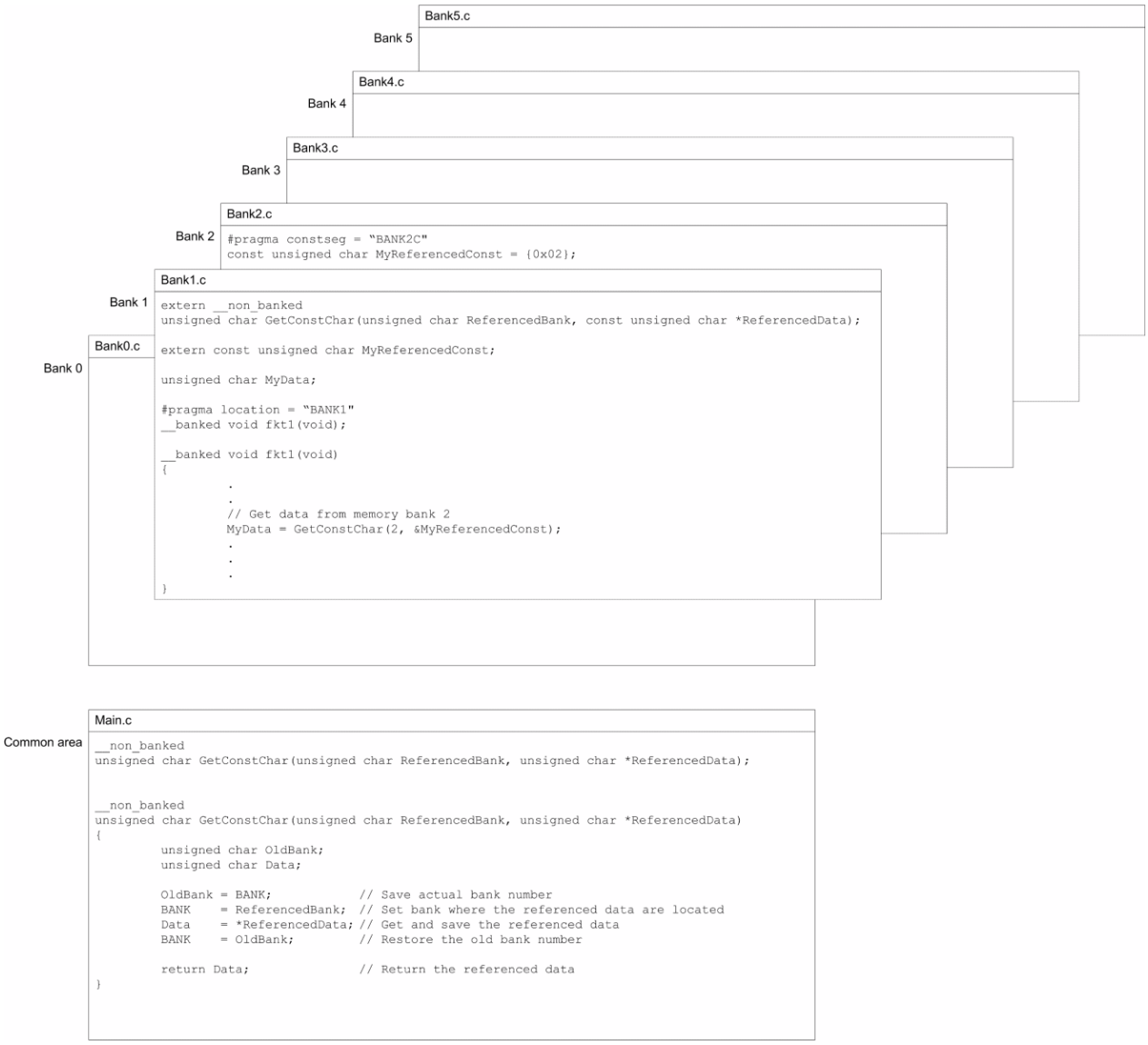
2.3.1 Referencing values between memory banks

Values cannot be directly referenced from one memory bank to another.

To access another memory bank from one memory bank, branch once to the common area (0000H to 7FFFH), change the setting of the BANK register there, and then reference a value.



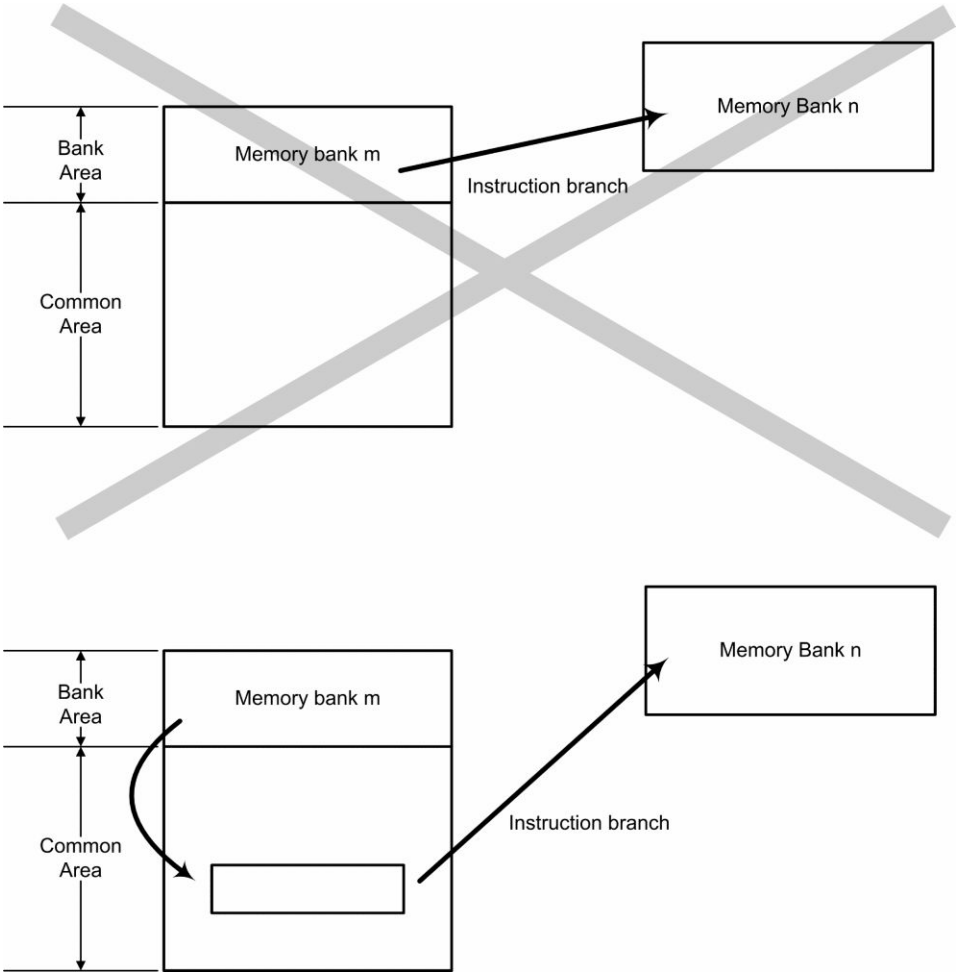
Software example (to read a constant located on memory bank 2 referenced from memory bank 1):



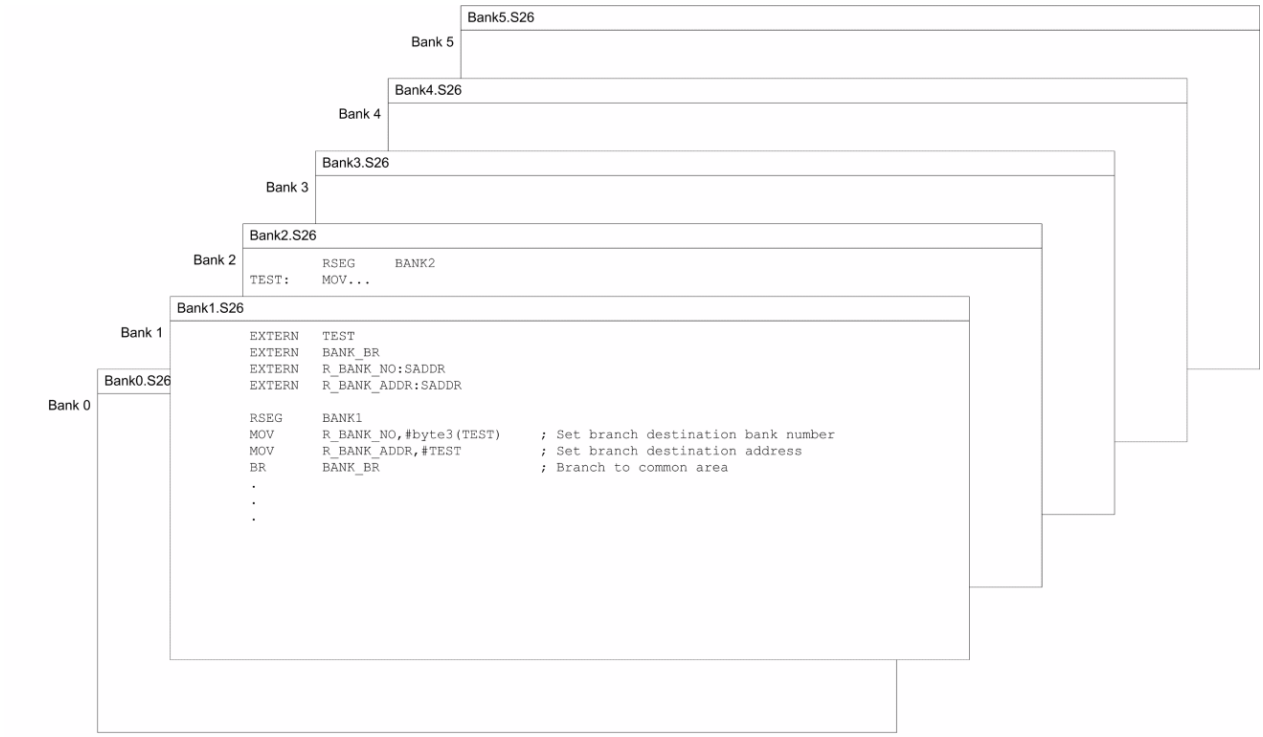
2.3.2 Instruction Branch Between Memory Banks

Instructions cannot branch directly from one memory bank to another.

To branch an instruction from one memory bank to another, branch once to the common area (0000H to 7FFFH), change the setting of the BANK register there, and then execute the branch instruction again.



Software example (to branch between memory banks)

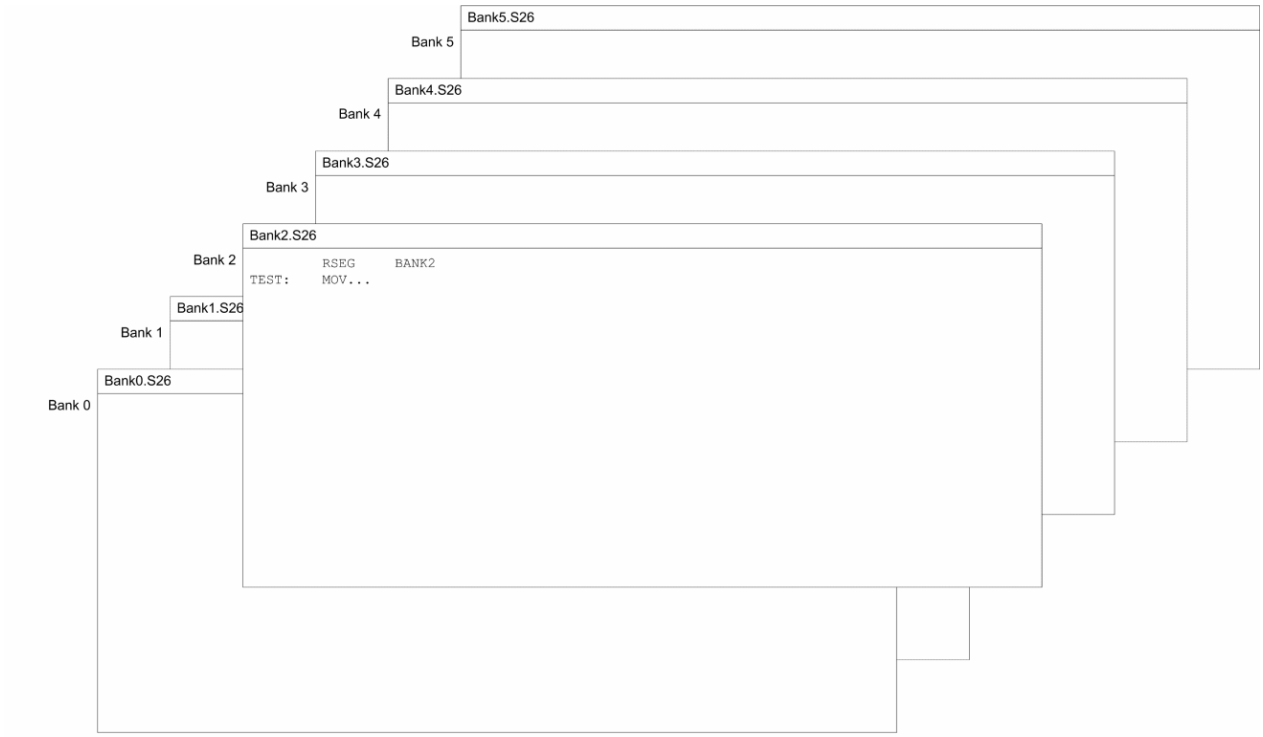


Common area

Main.S26

```
          RSEG    SADDR_Z
R_BANK_NO:
          DS      1                       ; Holds branch destination memory bank
R_BANK_ADDR:
          DS      2                       ; Holds branch destination address
R_SAVE_AX:
          DS      2                       ; Saves content of AX register

          RSEG    CODE
BANK_BR:  MOVW    R_SAVE_AX,AX            ; Save AX register
          MOV     A,R_BANK_NO             ; Set branch destination memory bank
          MOV     BANK,A
          MOVW    AX,R_BANK_ADDR          ; Save branch destination address on stack
          PUSH    AX
          MOVW    AX,R_SAVE_AX            ; Restore AX register
          RET                             ; Branch
```

Software example (to branch from the common area to any memory bank)



Common area — Main.S26

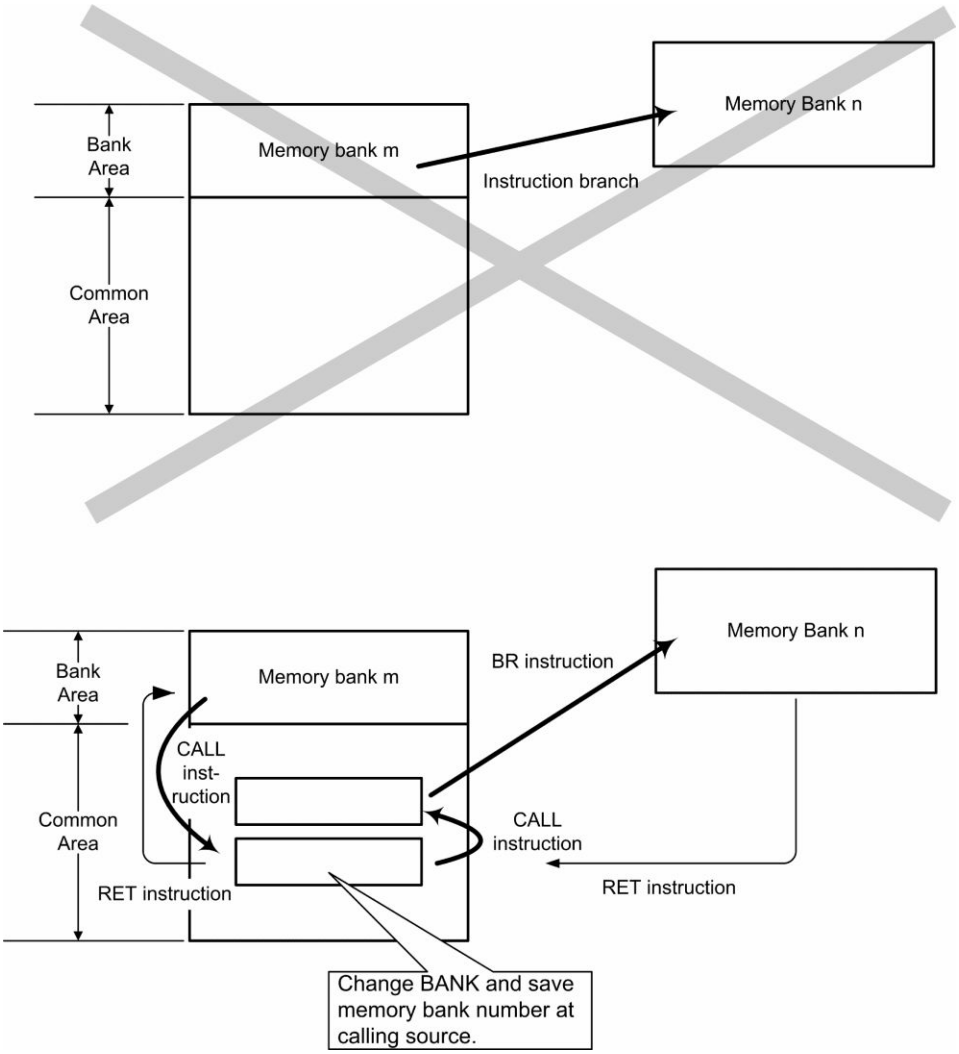
```
        EXTERN  TEST

        RSEG    SADDR_Z
R_BANK_NO:
        DS      1                       ; Holds branch destination memory bank
R_BANK_ADDR:
        DS      2                       ; Holds branch destination address
R_SAVE_AX:
        DS      2                       ; Saves content of AX register

        RSEG    CODE
ENTRY:  MOV     R_BANK_NO,#byte3(TEST)  ; Set branch destination bank number
        BR      TEST                    ; Branch to destination address
```

2.3.3 Subroutine call between memory banks

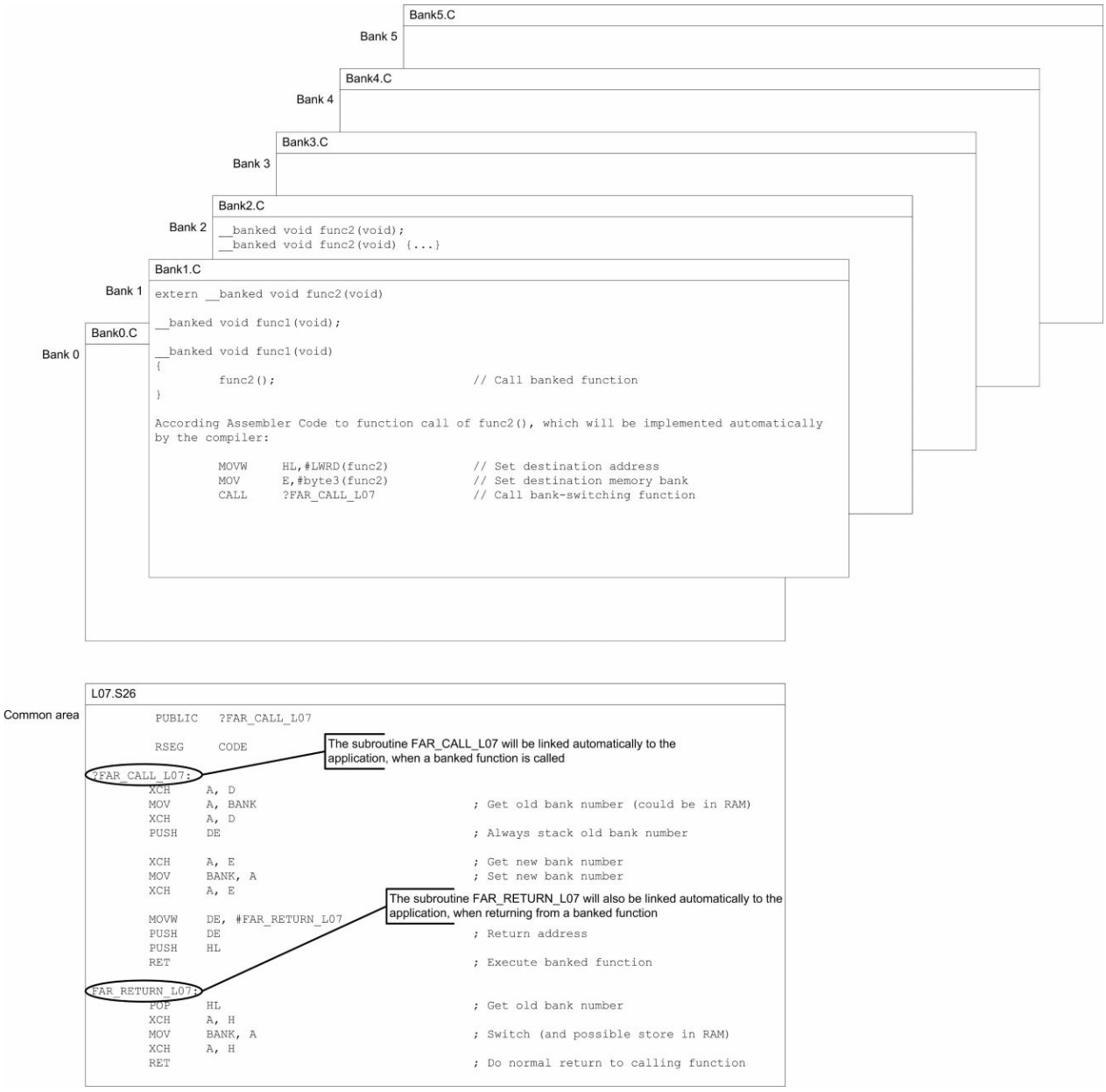
Subroutines cannot be directly called between memory banks.

To call a subroutine between memory banks, branch once to the common area (0000H to 7FFFH), specify the memory bank at the calling destination by using the BANK register there, execute the CALL instruction, and branch to the call destination by that instruction.

At this time, save the current value of the BANK register to RAM. Restore the value of the BANK register before executing the RET instruction.



Software example (to call a function on a memory bank from another memory bank)

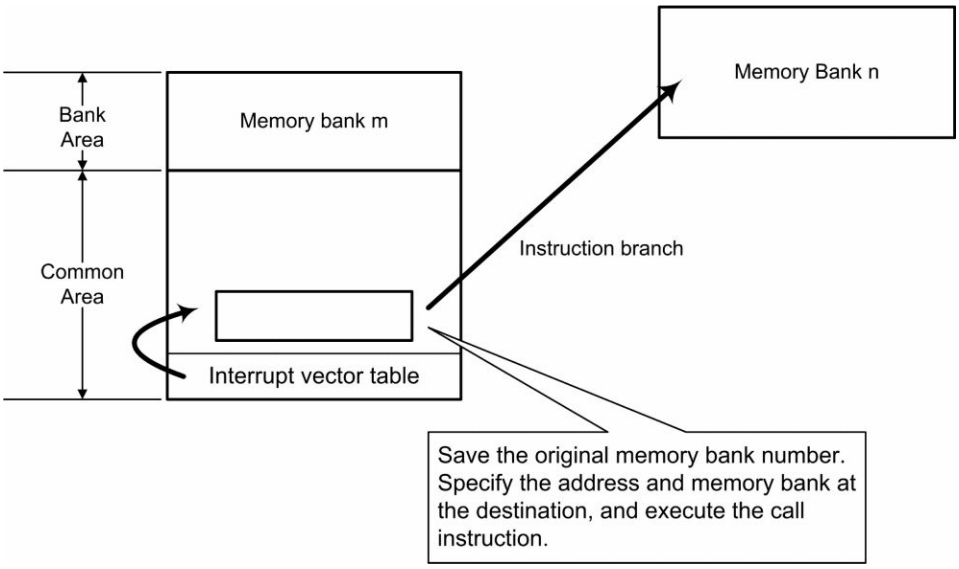


2.3.4 Instruction branch to bank area by interrupt

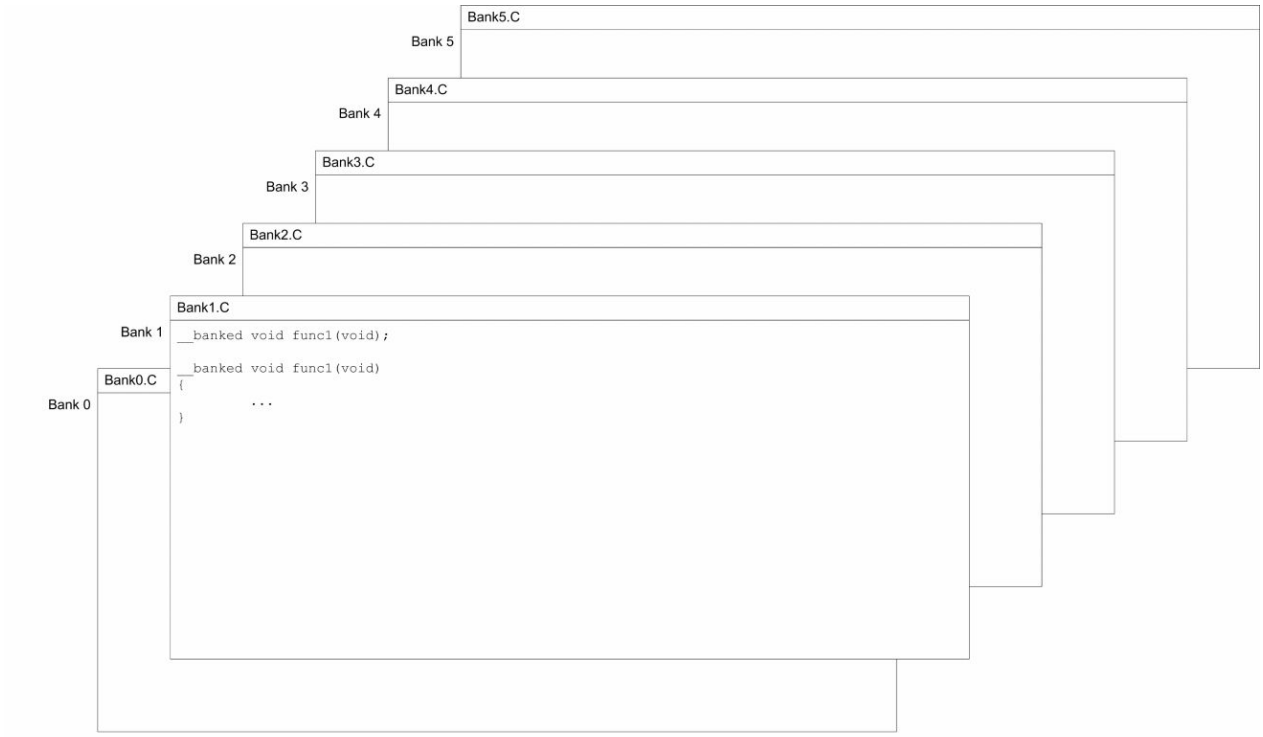
When an interrupt occurs, instructions can branch to the memory bank specified by the BANK register by using the vector table, but it is difficult to identify the BANK register when the interrupt occurs.

Therefore, specify the branch destination address specified by the vector table in the common area (0000H to 7FFFH), specify the memory bank at the branch destination by using the BANK register in the common area, and execute the CALL instruction.

At this time, save the BANK register value before the change to RAM, and restore the value of the BANK register before executing the RETI instruction.



Software example (to branch to any memory bank by interrupt)



Common area — L07.S26

```
        PUBLIC   ?FAR_CALL_L07

        RSEG     CODE

?FAR_CALL_L07:
        XCH      A, D
        MOV      A, BANK                        ; Get old bank number (could be in RAM)
        XCH      A, D
        PUSH     DE                             ; Always stack old bank number

        XCH      A, E                           ; Get new bank number
        MOV      BANK, A                        ; Set new bank number
        XCH      A, E

        MOVW     DE, #FAR_RETURN_L07
        PUSH     DE                             ; Return address
        PUSH     HL
        RET                                     ; Execute banked function

FAR_RETURN_L07:
        POP      HL                             ; Get old bank number
        XCH      A, H
        MOV      BANK, A                        ; Switch (and possible store in RAM)
        XCH      A, H
        RET                                     ; Do normal return to calling function
```

The subroutine FAR_CALL_L07 will be linked automatically to the application, when a banked function is called

The subroutine FAR_RETURN_L07 will also be linked automatically to the application, when returning from a banked function

Common area — Main.c

```
extern __banked void func1(void);

#pragma vector = INTxxx_vect
__interrupt void IsrINTxxx(void)
{
        func1();
}

According Assembler Code to function call of func1(), which will be implemented automatically
by the compiler:

        MOVW     HL,#LWRD(func1)          // Set destination address
        MOV      E,#byte3(func1)          // Set destination memory bank
        CALL     ?FAR_CALL_L07            // Call bank-switching function
```

Chapter 3 Linker Command File (*.XCL)

The IAR XLINK linker operation can be controlled by so-called XLINK options, which can be specified in the linker command (*.XCL) file.

In applications which are using the code expansion technique code banking, the linker has to be informed about the number and the size of the code memory banks.

This can be done in the linker command (*.XCL) file by setting the following options.

```
//-------------------------------------------------------------------------
//      Used with banked functions only
//      Start and end address of the code bank area.
//      Number of banks in the code bank area.
//      Remove comments and modify numbers if used from command line.
//-------------------------------------------------------------------------
-D_CODEBANK_START=8000
-D_CODEBANK_END=BFFF
-D_CODEBANK_BANKS=6

//-------------------------------------------------------------------------
//      Banked functions code segment.
//      The following code segments are available:
//      - BCODE segment uses all banks
//      - BANKx,BANKCx segments use only bank x
//
//CODE            Default segment for program code.
//BCODE           Holds all banked program code
//BANKn           List of defined banked code and banked const segments
//BANKCn
//_CODEBANK_START Start address of the banked area in hex
//_CODE_BANK_END  End address of the banked area in hex
//_CODE_BANKS     Decimal number of available banked segments
//-------------------------------------------------------------------------
-P(CODE)BCODE=[_CODEBANK_START-_CODEBANK_END]*_CODEBANK_BANKS+10000
-Z(CODE)BANK0,BANKC0=[(_CODEBANK_START+00000)-(_CODEBANK_END+00000)]
-Z(CODE)BANK1,BANKC1=[(_CODEBANK_START+10000)-(_CODEBANK_END+10000)]
-Z(CODE)BANK2,BANKC2=[(_CODEBANK_START+20000)-(_CODEBANK_END+20000)]
-Z(CODE)BANK3,BANKC3=[(_CODEBANK_START+30000)-(_CODEBANK_END+30000)]
-Z(CODE)BANK4,BANKC4=[(_CODEBANK_START+40000)-(_CODEBANK_END+40000)]
-Z(CODE)BANK5,BANKC5=[(_CODEBANK_START+50000)-(_CODEBANK_END+50000)]
//-------------------------------------------------------------------------
```

The above used XLINK options forces the IAR XLINK linker to use logical addresses for the code memory banks as described in column *Address Representation in Debugger of Table 2-1 on page 10* .

This means, that these XLINK options can only be used, if the Debugger in the target Debug of the IAR Embedded Workbench is used in order to create a C-SPY Debugger or NEC-Debugger output file.

In case that a HEX-file in the target Release of the IAR Embedded Workbench shall be generated in order to program a device with a flash programmer, the address representation must be changed to physical addresses as described in column *Flash Memory Real Address of Table 2-1 on page 10* .

Logical to physical address translation can be set in the linker command (*.XCL) file by adding the following options.

```
//-------------------------------------------------------------------------
//      Logical to physical address translation
//-------------------------------------------------------------------------
-M18000-1BFFF=0C000-0FFFF
```

```
-M28000-2BFFF=10000-13FFF
-M38000-3BFFF=14000-17FFF
-M48000-4BFFF=18000-1BFFF
-M58000-5BFFF=1C000-1FFFF
```

Note

1. NEC provides for all devices two linker command (*.XCL) files, one for use with the debugger and one for generation of the HEX-file. The linker command (*.XCL) file for HEX-file generation has the additional suffix HEX in the file name, e.g. DF054780_HEX_V4.XCL
2. Please use the following linker output file formats:
   - *xcoff78k* if the NEC Debugger is used, and
   - *intel-extended as segmented variant* if the real device shall be programmed with a flash programmer.
   - The linker output file format for the IAR C-SPY Debugger is selected automatically, if the IAR Embedded Workbench editor is used.

Revision History

Chapter	Page	Description
3	21	Logical to physical address translation table has been revised.