

## 7542 Group

### QzROM Multiple-Numbered Programming Application (2)

---

#### 1. Abstract

The following document describes QzROM Multiple-Numbered Programming Application in the 7542 group.

#### 2. Introduction

The application example described in this document is applied to the following conditions:

Applicable MCU: 7542QzROM(16K)

In this sample program, the bit in the unused function may be operated depending on the bit assignment of SFR. Setting values depend on a user system.

7542 QzROM version is a product in planning.  
Specifications or part numbers of a microcomputer may be changed.

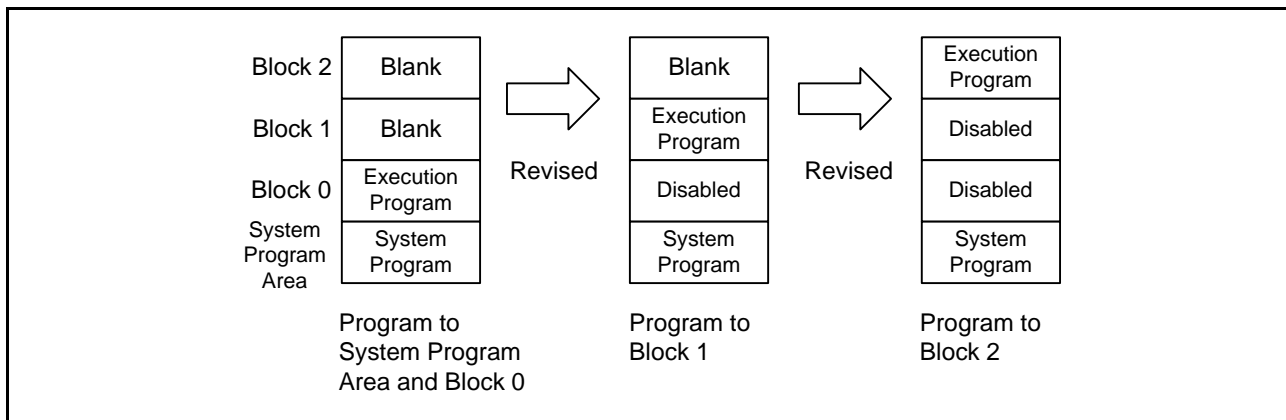
### 3. Description of Application Examples

#### 3.1 Target

In the QzROM version, additionally programming to blank area is enabled although reprogramming is disabled. In this application note, to enable a program updated in the QzROM version using this additional programming will be a goal.

In this application example, a program with capacity 4KB or below can be written 3 times using QzROM with ROM capacity 16KB. When the revised program is additionally written, the previous program will be disabled and a new written program will be operated.

Since the program to operate is selected by a start-up program, H/W such as external switch and so on, is not necessary.



**Figure 3.1 Additional Programming to Each Block and Execution Program**

## 3.2 Program Overview

In the 7542 group, the interrupt vector address is fixed in ROM area.

Therefore, when modifying the execution program, set the followings in advance, separately from a user program for an execution address after modification. (1), (2) and (4) are used after the execution program is modified. Refer to Figure 3.3, Memory Map for address of each area.

- (1) Start-up process : Start from this process after reset
- (2) System interrupt process : An interrupt process designated at the interrupt vector address. A program jumps to each user interrupt address designated at the user vector RAM indirectly.
- (3) User interrupt vector : The starting address of user program interrupt process is set in each block as a user interrupt vector. (See Figure 3.2).
- (4) User vector RAM : Each user interrupt address which jumps indirectly from the start-up process and system interrupt process is stored.

Figure 3.2 shows the Program Flow and How to Set Vector.

After reset is deasserted, execute the start-up process. When the execution block is determined by the block search, the user interrupt vector of the block is copied to the user vector RAM. And then, the user program main process will be executed by jumping indirectly to the user RESET interrupt address stored in the user vector RAM. When an interrupt is generated, the interrupt process is executed by jumping indirectly to the starting address of the user interrupt process stored in the user vector RAM with the system interrupt process.

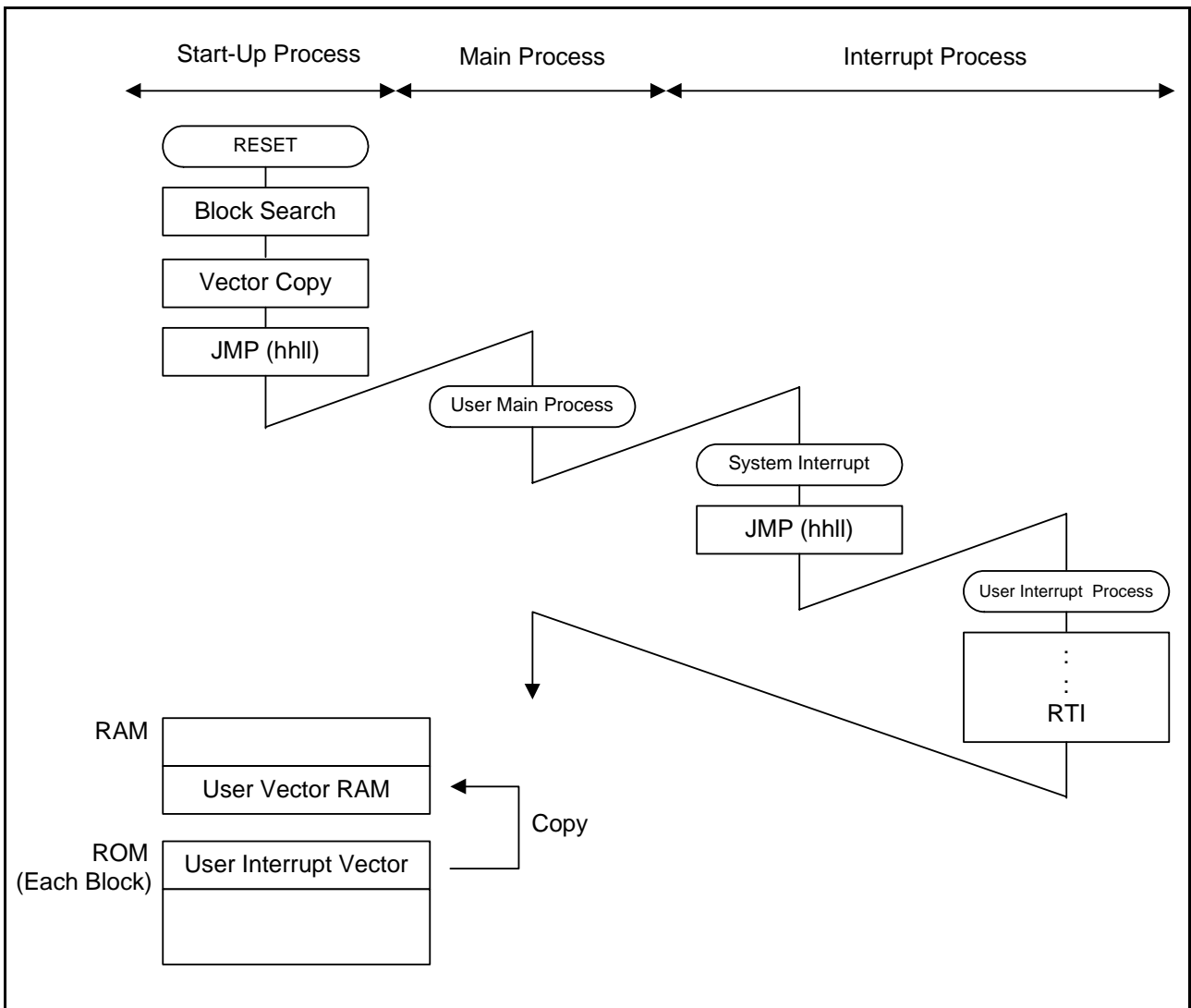


Figure 3.2 Program Flow and How to Set Vector

How to search block

The operation procedure to use blocks should be 0, 1 and 2.

The block search executes a blank check of the user RESET interrupt address in order of 2, 1 and 0.

If the address is not blank (FF16), the program identifies the checked block has the updated program.

### 3.3 Memory Map

Figure 3.3 shows the Memory Map, Figure 3.4 shows the User Interrupt Vector Area, Figure 3.5 shows the User Vector RAM Area, Figure 3.6 shows the Memory Size to be Used, Figure 3.7 shows the Flow Chart of Start-Up Process, Figure 3.8 shows the Example of User Program, Figure 3.9 shows the Example of System Interrupt Process and Figure 3.10 shows the Example of User Program (Interrupt Process).

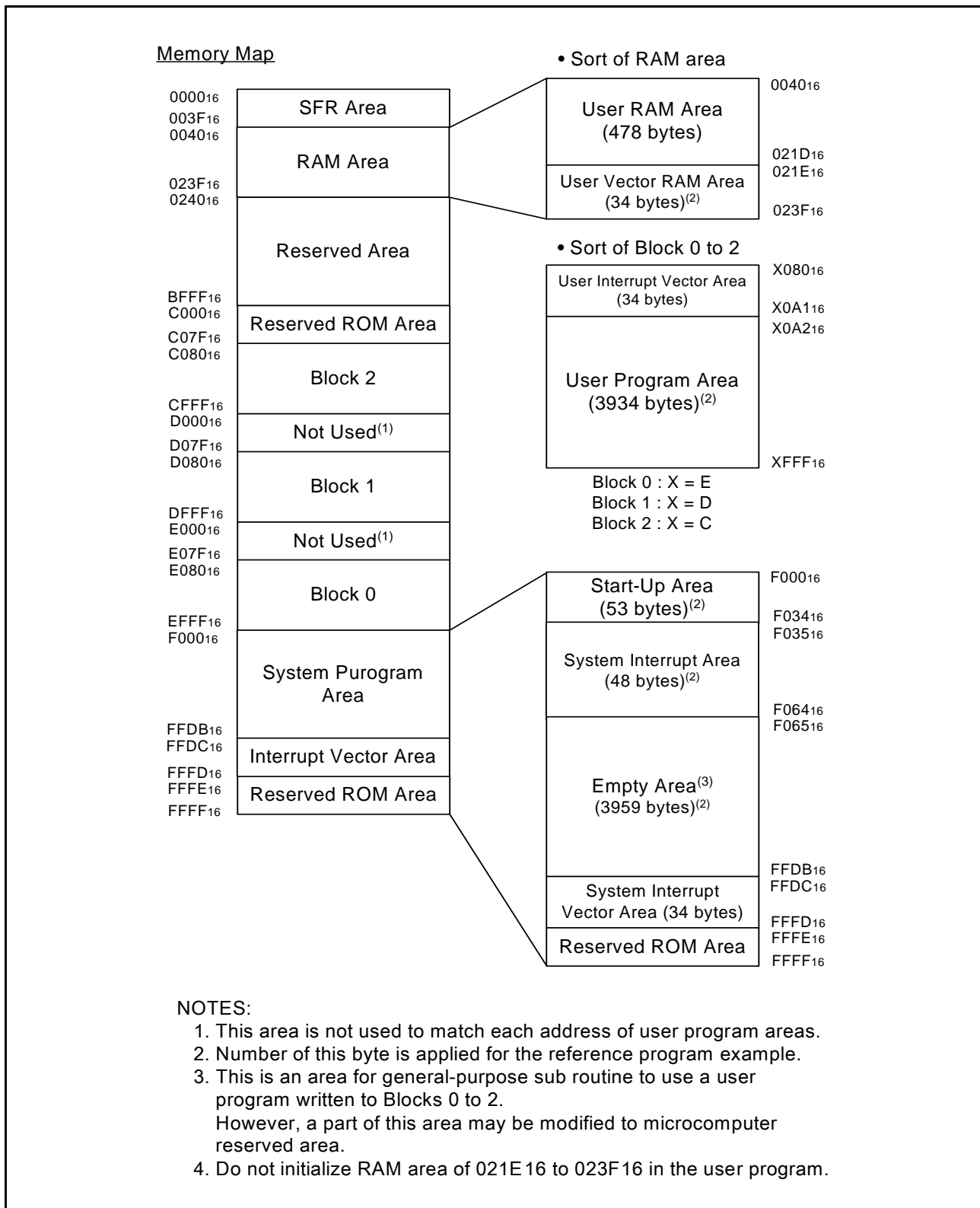


Figure 3.3 Memory Map

### User Interrupt Vector Area

User interrupt vector area is allocated at the beginning of each block (X080<sub>16</sub> to X0A1<sub>16</sub>), and this area is to set the starting address of the user program interrupt process.

Address	Stored Contents
X080 <sub>16</sub>	User BRK Instruction Interrupt Address (low-order)
X081 <sub>16</sub>	User BRK Instruction Interrupt Address (high-order)
X082 <sub>16</sub>	User AD / Timer 1 Interrupt Address (low-order)
X083 <sub>16</sub>	User AD / Timer 1 Interrupt Address (high-order)
X084 <sub>16</sub>	User Timer B Interrupt Address (low-order)
X085 <sub>16</sub>	User Timer B Interrupt Address (high-order)
X086 <sub>16</sub>	User Timer A Interrupt Address (low-order)
X087 <sub>16</sub>	User Timer A Interrupt Address (high-order)
X088 <sub>16</sub>	User Timer X Interrupt Address (low-order)
X089 <sub>16</sub>	User Timer X Interrupt Address (high-order)
X08A <sub>16</sub>	User Compare Interrupt Address (low-order)
X08B <sub>16</sub>	User Compare Interrupt Address (high-order)
X08C <sub>16</sub>	User Capture 1 Interrupt Address (low-order)
X08D <sub>16</sub>	User Capture 1 Interrupt Address (high-order)
X08E <sub>16</sub>	User Capture 0 Interrupt Address (low-order)
X08F <sub>16</sub>	User Capture 0 Interrupt Address (high-order)
X090 <sub>16</sub>	User CNTR0 Interrupt Address (low-order)
X091 <sub>16</sub>	User CNTR0 Interrupt Address (high-order)
X092 <sub>16</sub>	User Key-On Wake-Up / UART1 Bus Conflict Detection Interrupt Address (low-order)
X093 <sub>16</sub>	User Key-On Wake-Up / UART1 Bus Conflict Detection Interrupt Address (high-order)
X094 <sub>16</sub>	User INT1 Interrupt Address (low-order)
X095 <sub>16</sub>	User INT1 Interrupt Address (high-order)
X096 <sub>16</sub>	User INT0 Interrupt Address (low-order)
X097 <sub>16</sub>	User INT0 Interrupt Address (high-order)
X098 <sub>16</sub>	User Serial I/O2 Transmit Interrupt Address (low-order)
X099 <sub>16</sub>	User Serial I/O2 Transmit Interrupt Address (high-order)
X09A <sub>16</sub>	User Serial I/O2 Receive Interrupt Address (low-order)
X09B <sub>16</sub>	User Serial I/O2 Receive Interrupt Address (high-order)
X09C <sub>16</sub>	User Serial I/O1 Transmit Interrupt Address (low-order)
X09D <sub>16</sub>	User Serial I/O1 Transmit Interrupt Address (high-order)
X09E <sub>16</sub>	User Serial I/O1 Receive Interrupt Address (low-order)
X09F <sub>16</sub>	User Serial I/O1 Receive Interrupt Address (high-order)
X0A0 <sub>16</sub>	User RESET Interrupt Address (low-order)
X0A1 <sub>16</sub>	User RESET Interrupt Address (high-order)

Block 0 : X = E

Block 1 : X = D

Block 2 : X = C

**Figure 3.4 User Interrupt Vector Area**

### User Vector RAM Area

User vector RAM area is allocated at 021E<sub>16</sub> to 023F<sub>16</sub> in RAM area and a jump address to jump to the user interrupt process when a system interrupt is processed. The value in this area is copied to user interrupt vector area of the execution block with a start-up process.

Address	Symbol Name	Stored Contents
021E <sub>16</sub>	_RAM_BRK	User BRK Instruction Interrupt Address (low-order)
021F <sub>16</sub>		User BRK Instruction Interrupt Address (high-order)
0220 <sub>16</sub>	_RAM_AD	User AD / Timer 1 Interrupt Address (low-order)
0221 <sub>16</sub>		User AD / Timer 1 Interrupt Address (high-order)
0222 <sub>16</sub>	_RAM_TIMERB	User Timer B Interrupt Address (low-order)
0223 <sub>16</sub>		User Timer B Interrupt Address (high-order)
0224 <sub>16</sub>	_RAM_TIMERA	User Timer A Interrupt Address (low-order)
0225 <sub>16</sub>		User Timer A Interrupt Address (high-order)
0226 <sub>16</sub>	_RAM_TIMERX	User Timer X Interrupt Address (low-order)
0227 <sub>16</sub>		User Timer X Interrupt Address (high-order)
0228 <sub>16</sub>	_RAM_COMPARE	User Compare Interrupt Address (low-order)
0229 <sub>16</sub>		User Compare Interrupt Address (high-order)
022A <sub>16</sub>	_RAM_CAPTURE1	User Capture 1 Interrupt Address (low-order)
022B <sub>16</sub>		User Capture 1 Interrupt Address (high-order)
022C <sub>16</sub>	_RAM_CAPTURE0	User Capture 0 Interrupt Address (low-order)
022D <sub>16</sub>		User Capture 0 Interrupt Address (high-order)
022E <sub>16</sub>	_RAM_CNTR0	User CNTR0 Interrupt Address (low-order)
022F <sub>16</sub>		User CNTR0 Interrupt Address (high-order)
0230 <sub>16</sub>	_RAM_KEY	User Key-On Wake-Up / UART1 Bus Conflict Detection Interrupt Address (low-order)
0231 <sub>16</sub>		User Key-On Wake-Up / UART1 Bus Conflict Detection Interrupt Address (high-order)
0232 <sub>16</sub>	_RAM_INT1	User INT1 Interrupt Address (low-order)
0233 <sub>16</sub>		User INT1 Interrupt Address (high-order)
0234 <sub>16</sub>	_RAM_INT0	User INT0 Interrupt Address (low-order)
0235 <sub>16</sub>		User INT0 Interrupt Address (high-order)
0236 <sub>16</sub>	_RAM_SIO2T	User Serial I/O2 Transmit Interrupt Address (low-order)
0237 <sub>16</sub>		User Serial I/O2 Transmit Interrupt Address (high-order)
0238 <sub>16</sub>	_RAM_SIO2R	User Serial I/O2 Receive Interrupt Address (low-order)
0239 <sub>16</sub>		User Serial I/O2 Receive Interrupt Address (high-order)
023A <sub>16</sub>	_RAM_SIO1T	User Serial I/O1 Transmit Interrupt Address (low-order)
023B <sub>16</sub>		User Serial I/O1 Transmit Interrupt Address (high-order)
023C <sub>16</sub>	_RAM_SIO1R	User Serial I/O1 Receive Interrupt Address (low-order)
023D <sub>16</sub>		User Serial I/O1 Receive Interrupt Address (high-order)
023E <sub>16</sub>	_RAM_RESET	User RESET Interrupt Address (low-order)
023F <sub>16</sub>		User RESET Interrupt Address (high-order)

Figure 3.5 User Vector RAM Area

### Memory Size to be Used

Memory	Size	Remarks
Start-Up Area	53 bytes	Start-Up Process
System Interrupt Area	48 bytes	System Interrupt Process (3 bytes x 16 vectors)
User Vector RAM Area	34 bytes	021E <sub>16</sub> to 023F <sub>16</sub> (2 bytes x 17 vectors)
User Interrupt Vector Area	34 bytes	2 bytes x 17 vectors
User Program Area	3934 bytes	X0A2 <sub>16</sub> to XFFF <sub>16</sub> (X : C, D and E)
User RAM Area	478 bytes	0040 <sub>16</sub> to 021D <sub>16</sub>

Figure 3.6 Memory Size to be Used



## 3.4 Example of Control Procedure

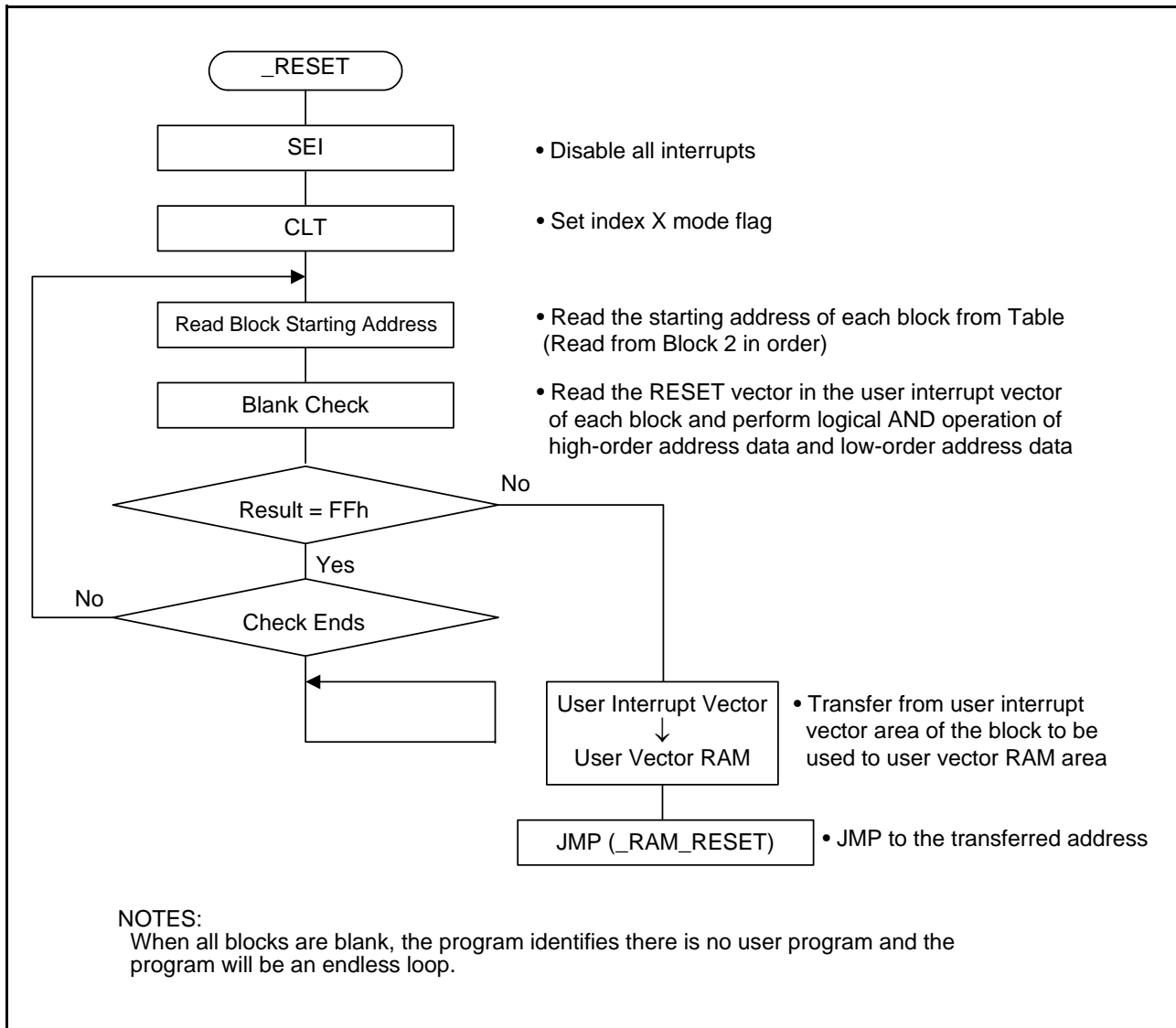


Figure 3.7 Flow Chart of Start-Up Process

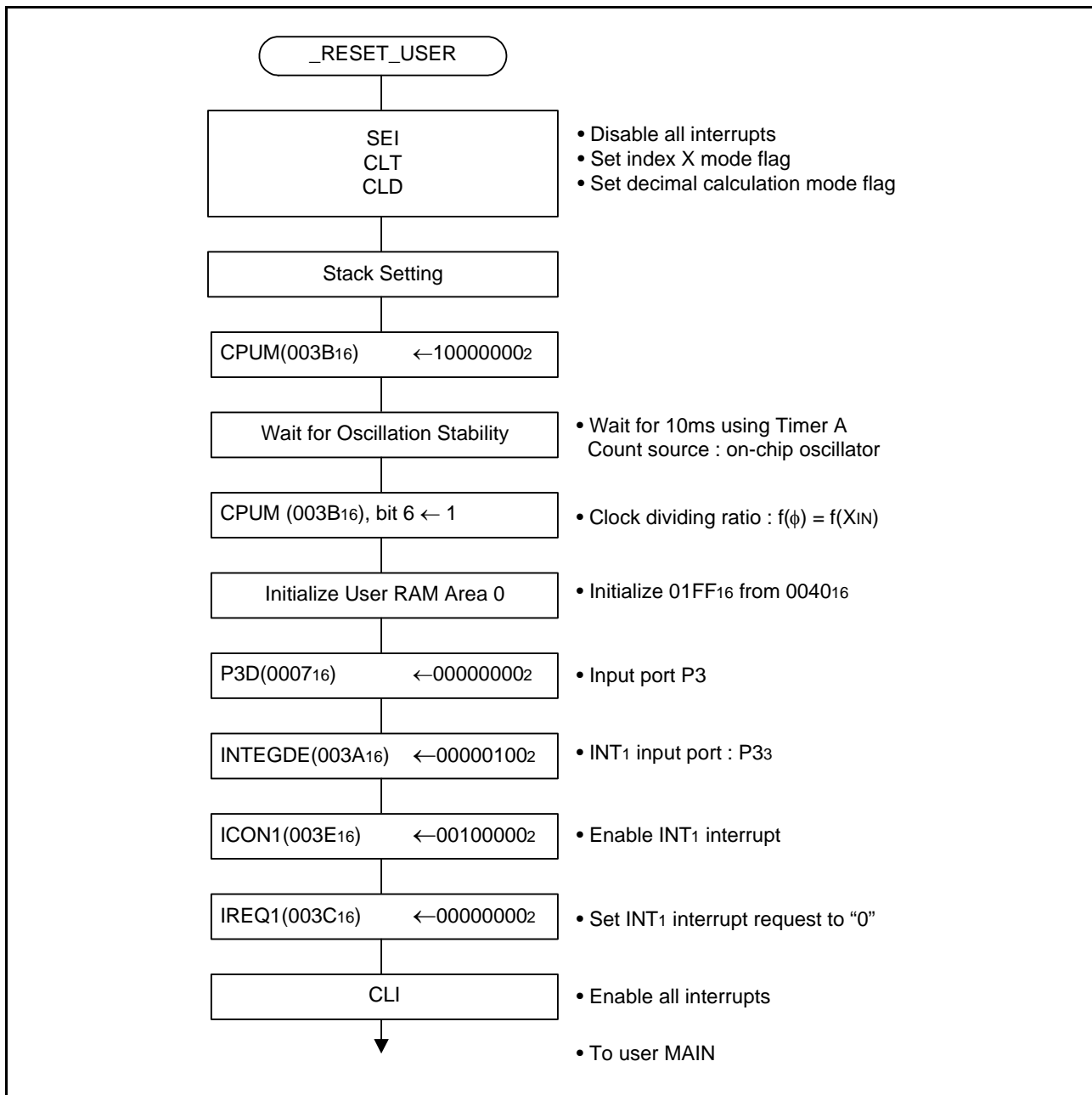


Figure 3.8 Example of User Program

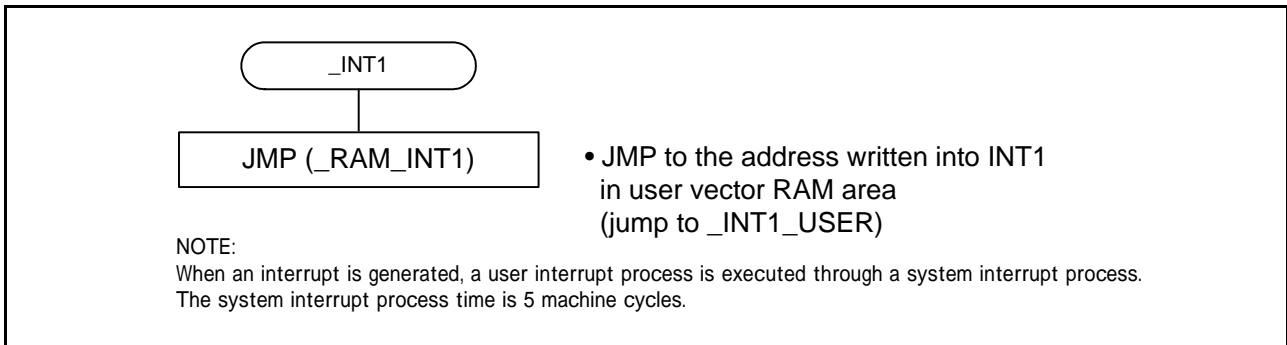


Figure 3.9 Example of System Interrupt Process

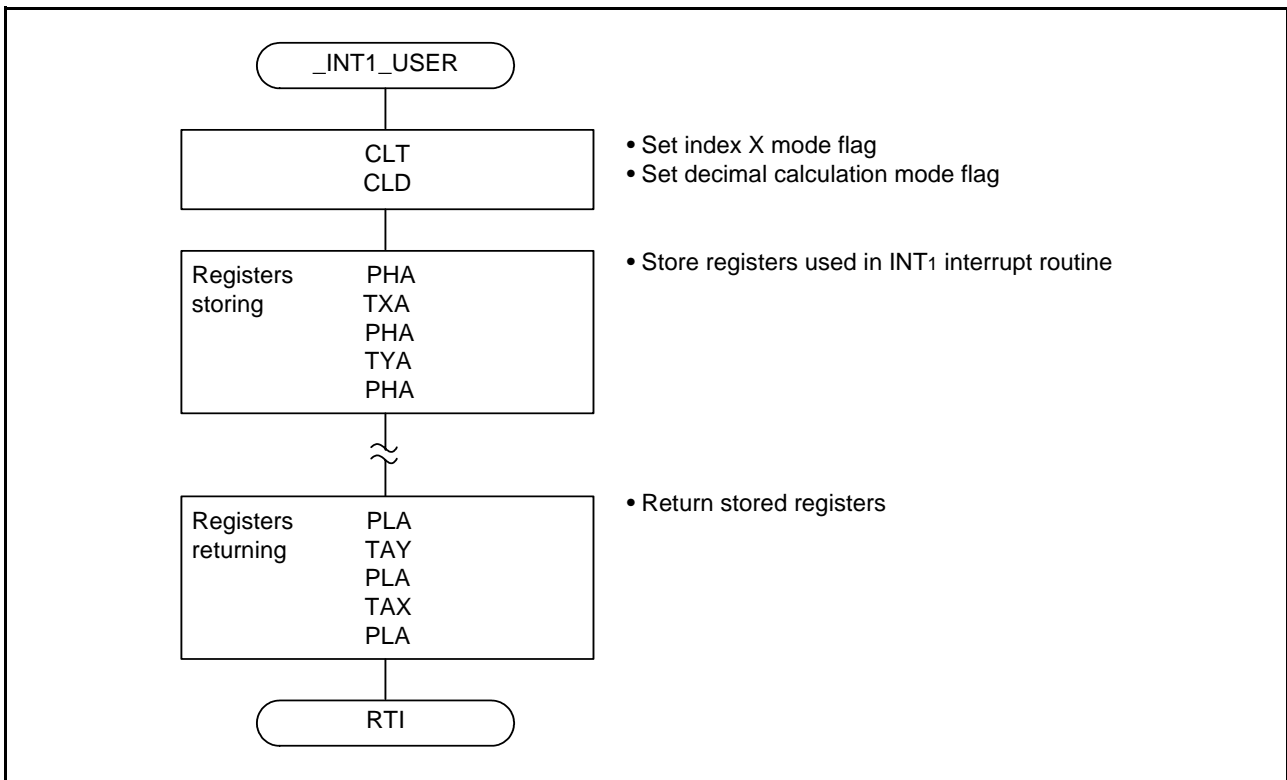


Figure 3.10 Example of User Program (Interrupt Process)

#### 4. Reference Program

```

[Block Starting Address Table]
.section __startup
.org    $0F000
__TBL_BLOCK_L:
.BYTE  >$C080           ;BLOCK2 start address(low)
.BYTE  >$D080           ;BLOCK1 start address(low)
.BYTE  >$E080           ;BLOCK0 start address(low)
;
__TBL_BLOCK_H:
.BYTE  <$C080           ;BLOCK2 start address(high)
.BYTE  <$D080           ;BLOCK1 start address(high)
.BYTE  <$E080           ;BLOCK0 start address(high)

[Start-Up Process]
__RESET:
SEI                ;Interrupt disable
CLT
;
; <<< search block >>>
LDX    #0
BLOCK_search:
LDA    __TBL_BLOCK_L,X
STA    $0040+0
LDA    __TBL_BLOCK_H,X
STA    $0040+1
LDY    #32           ;Set Index Y(user RESET vector)
LDA    ($0040),Y
INY
AND    ($0040),Y
CMP    #$FF           ;ROM = BLANK?
BNE    __VECTOR_COPY ;no >>
INX
CPX    #3           ;block search end?
BCC    BLOCK_search ;no >>
__VECTOR_ERR:
BRA    __VECTOR_ERR
;
; <<< copy user-vector >>>
__VECTOR_COPY:
LDY    #0
__VECTOR_COPY_LOOP:
LDA    ($0040),Y
STA    __RAM_BRK,Y
INY
CPY    #2*17         ;copy end?
BCC    __VECTOR_COPY_LOOP ;no >>
;
; <<< jump user reset>>>
JMP    (__RAM_RESET)

```

```

[System Interrupt Process]
__BRK:
    JMP    (__RAM_BRK)
;
__AD:
    JMP    (__RAM_AD)
;
__TIMERB:
    JMP    (__RAM_TIMERB)
;
__TIMERA:
    JMP    (__RAM_TIMERA)
;
__TIMERX:
    JMP    (__RAM_TIMERX)
;
__COMPARE:
    JMP    (__RAM_COMPARE)
;
__CAPTURE1:
    JMP    (__RAM_CAPTURE1)
;
__CAPTURE0:
    JMP    (__RAM_CAPTURE0)
;
__CNTR0:
    JMP    (__RAM_CNTR0)
;
__KEY:
    JMP    (__RAM_KEY)
;
__INT1:
    JMP    (__RAM_INT1)
;
__INT0:
    JMP    (__RAM_INT0)
;
__SIO2T:
    JMP    (__RAM_SIO2T)
;
__SIO2R:
    JMP    (__RAM_SIO2R)
;
__SIO1T:
    JMP    (__RAM_SIO1T)
;
__SIO1R:
    JMP    (__RAM_SIO1R)

[System Interrupt Vector]
.section vector
.org    0FFDCH
.word   __BRK           ; FFDC : BRK
.word   __AD            ; FFDE : AD, Timer1
.word   __TIMERB       ; FFE0 : TimerB
.word   __TIMERA       ; FFE2 : TimerA
.word   __TIMERX       ; FFE4 : TimerX
.word   __COMPARE      ; FFE6 : Compare
.word   __CAPTURE1     ; FFE8 : Capture 1
.word   __CAPTURE0     ; FFEA : Capture 0
.word   __CNTR0        ; FFEC : CNTR0

```

```

.WORD  __KEY           ; FFEE : Key-on wake-up, UART1 bus collision detection
.WORD  __INT1          ; FFF0 : INT1
.WORD  __INT0          ; FFF2 : INT0
.WORD  __SIO2T         ; FFF4 : Serial I/O2 transmit
.WORD  __SIO2R         ; FFF6 : Serial I/O2 receive
.WORD  __SIO1T         ; FFF8 : Serial I/O1 transmit
.WORD  __SIO1R         ; FFFA : Serial I/O1 receive
.WORD  __RESET         ; FFFC : RESET

[User Interrupt Vector]
.section __USER_VECTOR
; ; .org $C080 ;use block2
; ; .org $D080 ;use block1
.org $E080 ;use block0
.WORD  DUMMY_INT      ; X080 : BRK
.WORD  DUMMY_INT      ; X082 : AD, Timer1
.WORD  DUMMY_INT      ; X084 : TimerB
.WORD  DUMMY_INT      ; X086 : TimerA
.WORD  DUMMY_INT      ; X088 : TimerX
.WORD  DUMMY_INT      ; X08A : Compare
.WORD  DUMMY_INT      ; X08C : Capture 1
.WORD  DUMMY_INT      ; X08E : Capture 0
.WORD  DUMMY_INT      ; X090 : CNTR0
.WORD  DUMMY_INT      ; X092 : Key-on wake-up, UART1 bus collision detection
.WORD  __INT1_USER    ; X094 : INT1
.WORD  DUMMY_INT      ; X096 : INT0
.WORD  DUMMY_INT      ; X098 : Serial I/O2 transmit
.WORD  DUMMY_INT      ; X09A : Serial I/O2 receive
.WORD  DUMMY_INT      ; X09C : Serial I/O1 transmit
.WORD  DUMMY_INT      ; X09E : Serial I/O1 receive
.WORD  __RESET_USER   ; X0A0 : RESET

[User Program]
.section P
__RESET_USER:
SEI           ;Interrupt disable
CLT
CLD

LDX  #$FF           ;Set stack bottom
TXS

LDM  #%10000000,CPUM ;Set CPU mode register
;
;Wait f(XIN)oscillation stabilizing time
;
SEB  6,CPUM         ;f(Xin)
;
; <<< RAM clear >>>
LDA  #0
LDX  #$40
__RAM_page0_clear:
STA  $0000,X
INX
BNE  __RAM_page0_clear
__RAM_page1_clear:
STA  $0100,X
INX
BNE  __RAM_page1_clear
__RAM_page2_clear:
STA  $0200,X
INX

```

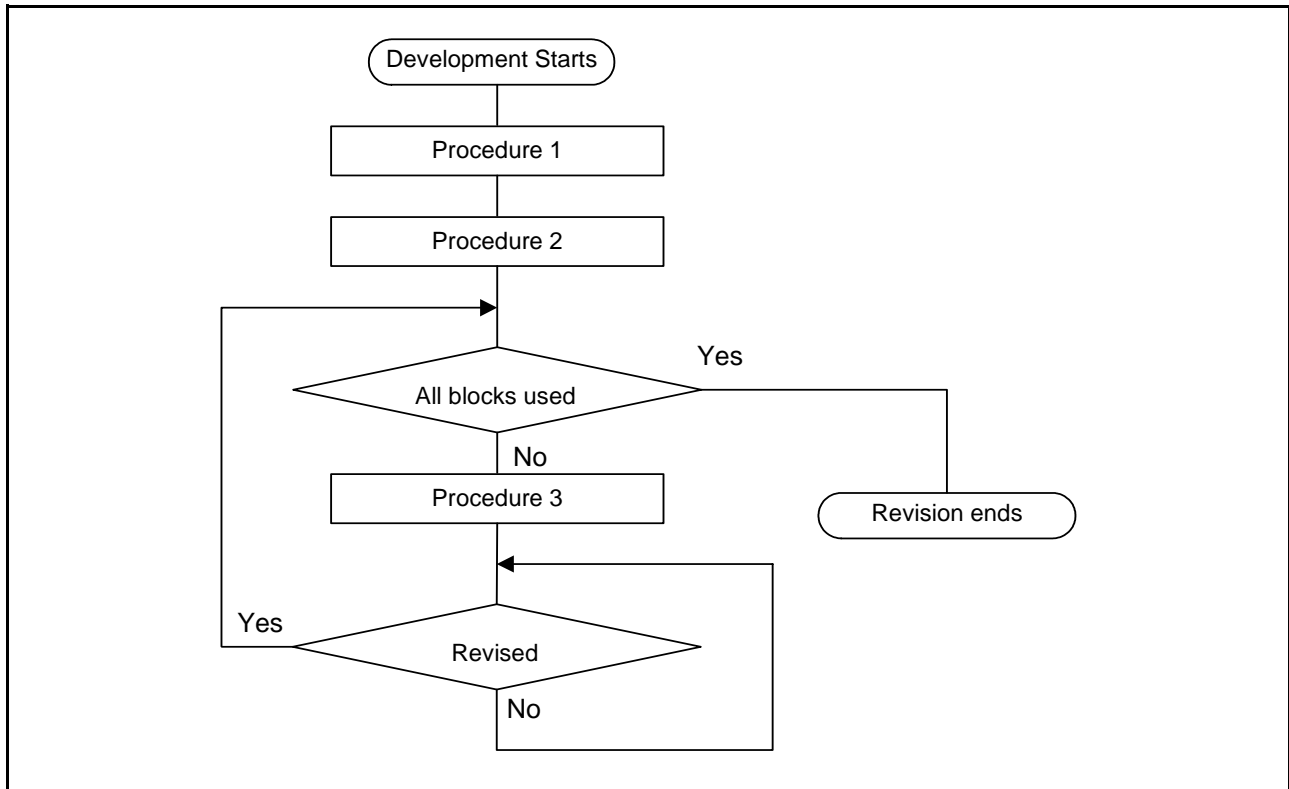


```
CPX    #$1E
BCC    __RAM_page2_clear
;
LDM    #%00000000,P3D    ;Set Port P3 direction register
LDM    #%00000100,INTEGE ;Set Interrupt edge selection register
LDM    #%00100000,ICON1  ;INT1 interrupt disable
LDM    #%00000000,IREQ1  ;INT1 interrupt request clear
;
CLI                    ;Interrupt enable
;
MAIN:
    BRA    MAIN

[User Interrupt Program]
__INT1_USER:
    CLT
    CLD

    PHA
    TXA
    PHA
    TYA
    PHA
;
    PLA
    TAY
    PLA
    TAX
    PLA
DUMMY_INT:
    RTI
```

## 5. Example of Program Development Procedure



**Figure 5.1 Development Procedure**

- Procedure 1  
A file for a start-up process and a file for a user program are assumed as separate files. The user program file can be divided. This application example consists of the following 2 files.  
Start-up process file : startup.a74  
User program file : user.a74
- Procedure 2  
Assemble only startup.a74  
Designate area of F00016 to FFFF16 and program to QzROM.



• Procedure 3

Program to Block 0

Designate the address declaration in the user program (user.a74) as below, designate areas of assemble E080<sub>16</sub> to EFFF<sub>16</sub> and program to the QzROM.

```
.section_USER_VECTOR
;;; .org    $C080    ;use block2
;;; .org    $D080    ;use block1
.org    $E080    ;use Block0
```

Program to Block 1

Designate the address declaration in the user program (user.a74) as below, designate areas of assemble D080<sub>16</sub> to DFFF<sub>16</sub> and program to the QzROM.

```
.section_USER_VECTOR
;;; .org    $C080    ;use block2
.org    $D080    ;use block1
;;; .org    $E080    ;use block0
```

Program to Block 2

Designate the address declaration in the user program (user.a74) as below, designate areas of assemble C080<sub>16</sub> to CFFF<sub>16</sub> and program to the QzROM.

```
.section_USER_VECTOR
.org    $C080    ;use block2
;;; .org    $D080    ;use block1
;;; .org    $E080    ;use block0
```

## 6. Reference

Data Sheet  
7542 Group Data sheet

Before using this material, please visit our website to verify that this is the most updated document available.

## 7. Home Page and Support

Renesas Technology Corporation Semiconductor Home Page  
<http://www.renesas.com>

E-mail Support  
E-mail: [support\\_apl@renesas.com](mailto:support_apl@renesas.com)

REVISION HISTORY	7542 Group QzROM Multiple-Numbered Programming Application (2)
------------------	--

Rev.	Date	Description	
		Page	Summary
0.10	Feb 01, 2005	-	Preliminary Edition issued

**Keep safety first in your circuit designs!**

1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of non-flammable material or (iii) prevention against any malfunction or mishap.

**Notes regarding these materials**

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.  
The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.  
Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (<http://www.renesas.com>).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.  
Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.