RX Family

Using Multiple Interrupts

R01AN1954EJ0100
Rev.1.00
Jul 01, 2014

Introduction

This application note describes how to use multiple interrupts on RX family microcontrollers.

Target Device

RX Family

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Contents

1. Specifications

While the IRQ1 interrupt is being processed, the IRQ3 interrupt occurs.

Normally, the I flag in PSW is cleared when an interrupt occurs, disabling other interrupts. In order to enable multiple interrupts, it is necessary to reset the I flag in PSW in the interrupt handler.

Setting the I flag to 1 enables interrupts with a higher priority level.

1.1 Settings to Enable Multiple Interrupts

It is necessary to use #pragma interrupt in the declarations of the interrupt function. This declaration specifies that multiple interrupts are enabled, allowing multiple interrupts to be generated at the same time. The following is an example multiple interrupt enable specification.

Example: IRQ1 interrupt function declaration on RX220
#pragma interrupt (Excep_IRQm (enable,vect = 65))

Once multiple interrupts have been enabled, it is possible to set the I flag in PSW to 1 at the beginning of the interrupt function to allow multiple interrupts.

For further details regarding #pragma interrupt, see the latest version of RX Family C/C++ Compiler Package User's Manual.

1.2 Peripheral Functions and Applications

Table 1.1 lists the peripheral functions used and their applications, and figure 1.1 is a connection diagram.

Table 1.1 Peripheral Functions and Applications

Peripheral Function	Application
IRQ	External interrupt
I/O port	LED illumination

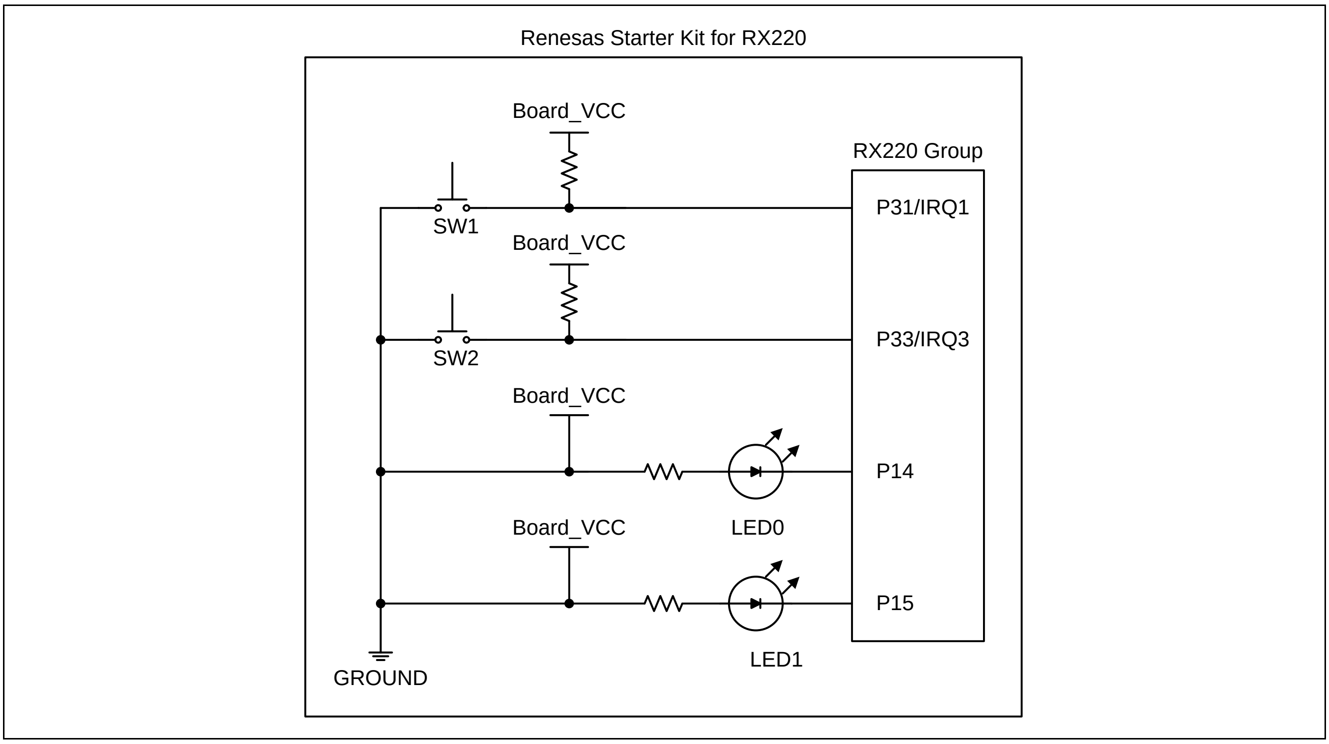


Figure 1.1 Connection Diagram

2. Operation Confirmation Conditions

The sample code accompanying this application note has been run and confirmed under the conditions below.

Table 2.1 Operation Confirmation Conditions

Item	Contents
MCU used	R5F52206BDFP (RX220 Group)
Operating frequency	Main clock: 20.0 MHz System clock (ICLK): 20 MHz (main clock divided by 1) Peripheral module clock B (PCLKB): 20 MHz (main clock divided by 1)
Operating voltage	5.0 V (supplied by E1 emulator)
Integrated development environment	Renesas electronics High-performance Embedded Workshop Version 4.09.01.007
C compiler	Renesas electronics C/C++ Compiler Package for RX Family V.1.02 Release 01
	Compiler option -cpu=rx200 -output=obj="$(CONFIGDIR)\$(FILELEAF).obj" -debug –nologo (The integrated development environment default settings are used.)
iodefine.h version	Version 1.0A
Endian order	Little endian
Operating mode	Single-chip mode
Processor mode	Supervisor mode
Sample code version	Version 1.00
Board used	Renesas Starter Kit for RX220 (Product number: R0K505220S000BE)

3. Reference Application Note

For additional information associated with this document, refer to the following application note.

- RX220 Group Initial Setting Rev.1.00 (R01AN1494EJ0100_RX220)

The initial settings function of the above application note is used in the sample code accompanying this application note. The revision number shown is that current at the time this application note was issued.

If a newer version is available, replace the current version with the newer version. The latest version can be confirmed and downloaded from the Renesas Electronics website.

4. Hardware

4.1 Used Pins

Table 4.1 lists the pins and their functions.

Table 4.1 Used Pins and Their Functions

Pin Name	I/O	Function
P14	Output	LED0 output
P15	Output	LED1 output
P31/IRQ1	Input	IRQ1 interrupt SW1 input
P33/IRQ3	Input	IRQ3 interrupt SW2 input

5. Software

5.1 Operation Overview

(1) Initial settings

After making initial settings for the clocks, I/O ports, and IRQ interrupts, the software waits for SW1 input. The function described in the application note RX Group: Initial Settings Example is called to make clock settings. The IRQ detection settings consist of setting level detection for both IRQ1 and IRQ3, setting IRQ1 to interrupt priority level 1, and setting IRQ3 to interrupt priority level 2.

(2) SW1 input detection

When SW1 is pressed (low-level input on P31), an IRQ1 interrupt request is received.

(3) IRQ1 interrupt

The IRQ1 interrupt handler turns on LED0. Processing loops within the IRQ1 interrupt handler for as long as SW1 is held down. When SW1 is released, LED0 is turned off and the IRQ1 interrupt handler ends.

(4) SW2 input detection

In like manner to SW1 input detection, when SW2 is pressed (low-level input on P33), an IRQ3 interrupt request is received.

(5) IRQ3 interrupt

The IRQ3 interrupt handler turns on LED1. Processing loops within the IRQ3 interrupt handler for as long as SW2 is held down. When SW2 is released, LED1 is turned off and the IRQ3 interrupt handler ends.

To generate multiple interrupts, press SW2 while holding down SW1. Note that pressing SW1 while holding down SW2 does not generate multiple interrupts because the interrupt priority level of IRQ1 is lower that that of IRQ3.

Figure 5.1 is a timing chart showing the occurrence of multiple interrupts.

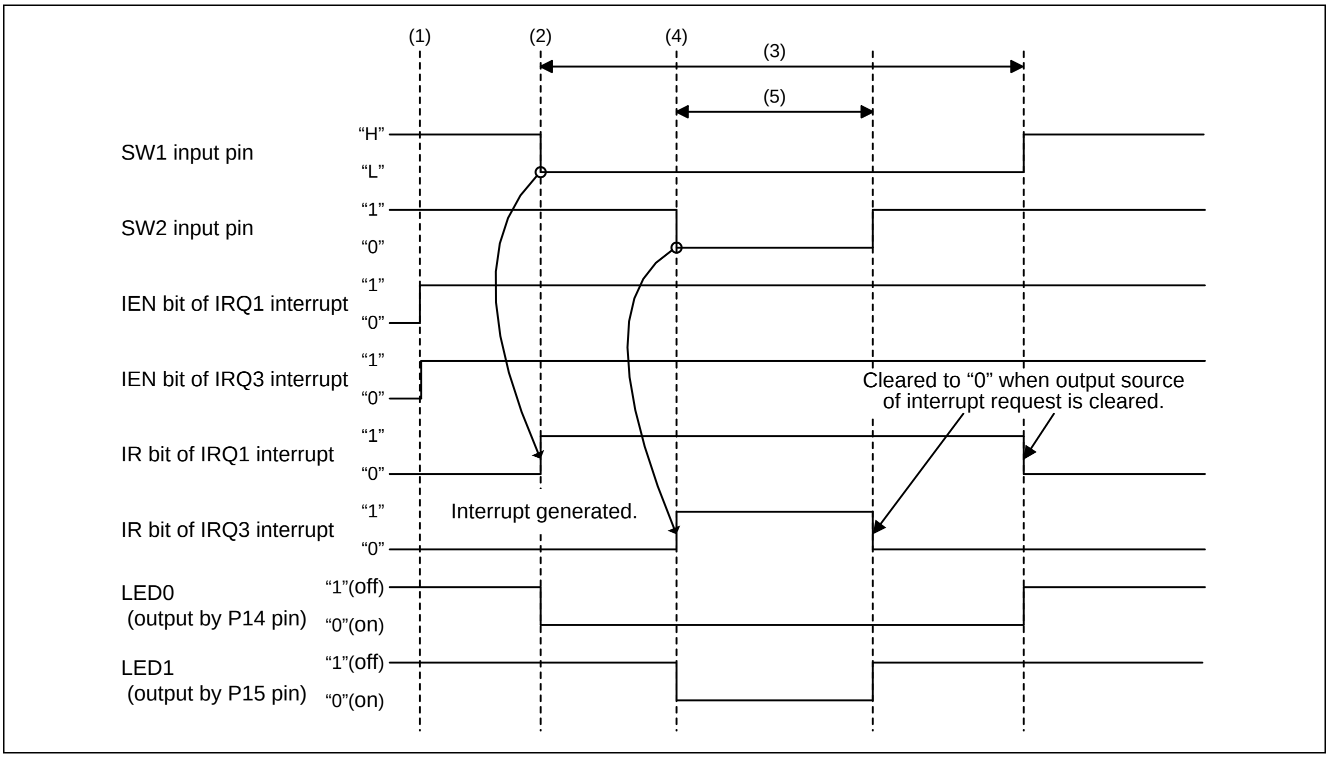


Figure 5.1 Timing Chart Showing Occurrence of Multiple Interrupts

5.2 File Composition

Table 5.1 lists the Files Used in the Sample Code. Files generated by the integrated development environment are not included in this table.

Table 5.1 Files Used in the Sample Code

File Name	Outline	Remarks
main.c	Main processing	RX family common settings
port_cfg.h	Port configuration header file	RX family common settings
rx220_port_init.c	Port and IRQ settings	RX220 port settings
r_init_stop_module.c	Stop peripheral functions operating after a reset	RX220 initial settings
r_init_stop_module.h	Header file for r_init_stop_module.c	RX220 initial settings
r_init_non_existent_port.c	Initial settings for unimplemented ports	RX220 initial settings
r_init_non_existent_port.h	Header file for r_init_non_existent_port.c	RX220 initial settings
r_init_clock.c	Initial clock settings	RX220 initial settings
r_init_clock.h	Header file for r_init_clock.c	RX220 initial settings

5.3 Option-Setting Memory

Table 5.2 lists the Option-Setting Memory Configured in the Sample Code. When necessary, set a value suited to the user system.

Table 5.2 Option-Setting Memory Configured in the Sample Code

Symbol	Address	Setting Value	Contents
OFS0	FFFF FF8Fh to FFFF FF8Ch	FFFF FFFFh	After a reset, the IWDT is stopped. After a reset, the WDT is stopped.
OFS1	FFFF FF8Bh to FFFF FF88h	FFFF FFFFh	After a reset, voltage monitoring reset 0 is ignored. Disables HOCO oscillation after a reset
MDES	FFFF FF83h to FFFF FF80h	FFFF FFFFh	Little endian

5.4 Constants

Table 5.3 and table 5.4 list the Constants Used in the Sample Code.

Table 5.3 Constants Used in the Sample Code (port_cfg.h)

Constant	Set Value	Description
LED_ON	0	LED output data: On
LED_OFF	1	LED output data: Off
IR_ICU_IRQm	IR(ICU,IRQ1)	IRQ1 interrupt status flag
IR_ICU_IRQn	IR(ICU,IRQ3)	IRQ3 interrupt status flag
IEN_ICU_IRQm	IEN(ICU,IRQ1)	IRQ1 interrupt request enable bit
IEN_ICU_IRQn	IEN(ICU,IRQ3)	IRQ3 interrupt request enable bit
IPR_ICU_IRQm	IPR(ICU,IRQ1)	IRQ1 interrupt priority level setting bit
IPR_ICU_IRQn	IPR(ICU,IRQ3)	IRQ3 interrupt priority level setting bit
IRQMD_ICU_IRQm	ICU.IRQCR[1].BYTE	IRQ1 detection setting bit
IRQMD_ICU_IRQn	ICU.IRQCR[3].BYTE	IRQ3 detection setting bit
SW1_REG_PDR	PORT3.PDR.BIT.B1	SW1 direction control bit
SW2_REG_PDR	PORT3.PDR.BIT.B3	SW2 direction control bit
LED0_REG_PODR	PORT1.PODR.BIT.B4	LED0 output data storage bit
LED1_REG_PODR	PORT1.PODR.BIT.B5	LED1 output data storage bit
LED0_REG_PDR	PORT1.PDR.BIT.B4	LED0 direction control bit
LED1_REG_PDR	PORT1.PDR.BIT.B5	LED1 direction control bit

Table 5.4 Constants Used in the Sample Code (rx220_port_init.c)

Constant	Set Value	Description
SW1_REG_PMR	PORT3.PMR.BIT.B1	SW1 pin mode control bit
SW2_REG_PMR	PORT3.PMR.BIT.B3	SW2 pin mode control bit
LED0_REG_PMR	PORT1.PMR.BIT.B4	LED0 pin mode control bit
LED1_REG_PMR	PORT1.PMR.BIT.B5	LED1 pin mode control bit
ISEL_MPC_PmPFS	MPC.P31PFS.BIT.ISEL	P31 interrupt input function select bit
ISEL_MPC_PnPFS	MPC.P33PFS.BIT.ISEL	P33 interrupt input function select bit

5.5 Functions

Table 5.5 lists the Functions.

Table 5.5 Functions

Function Name	Outline
main	Main processing
R_INIT_StopModule	Stop peripheral functions operating after a reset
R_INIT_NonExistentPort	Initial settings for unimplemented ports
R_INIT_Clock	Clock initial settings
port_init	Port initial settings
irq_init	IRQ initial settings
Excep_IRQm	IRQm interrupt handler (IRQ1 selected)
Excep_IRQn	IRQn interrupt handler (IRQ3 selected)

5.6 Function Specifications

The specifications of the sample code functions are shown below.

main	
Outline	Main processing
Header	None
Declaration	void main (void)
Description	Calls functions that make initial settings, port initial settings, and IRQ initial settings.
Arguments	None
Return Value	None

R_INIT_StopModule	
Outline	Stop peripheral functions operating after a reset
Header	r_init_stop_module.h
Declaration	void R_INIT_StopModule (void)
Description	Makes settings to transition to the module stop state.
Arguments	None
Return Value	In the sample code, no transition to the module stop state occurs. For detailed information on this function, see the application note RX Group: Initial Settings Example, Rev. 1.00.

R_INIT_NonExistentPort	
Outline	Initial settings for unimplemented ports
Header	r_init_non_existent_port.h
Declaration	void R_INIT_NonExistentPort (void)
Description	Makes initial settings to the port direction registers for ports corresponding to pins that are nor implemented on products with fewer than 100 pins.
Arguments	None
Return Value	In the sample code, the pin count is specified as 100 (PIN_SIZE=100). After this function is called, set the direction control bit to “1” and the port output data storage bit to “0” for unimplemented ports when writing in byte units to PDR and PODR registers that include unimplemented ports. For detailed information on this function, see the application note RX Group: Initial Settings Example, Rev. 1.00.

R_INIT_Clock	
Outline	Clock initial settings
Header	r_init_clock.h
Declaration	void R_INIT_Clock(void)
Description	Makes initial clock settings.
Arguments	None
Return Value	In the RX220 sample code, processing is selected in which the system clock is set as the main clock and no subclock is used. For detailed information on this function, see the application note RX Group: Initial Settings Example, Rev. 1.00.

port_init	
Outline	Port initial settings
Header	None
Declaration	void port_init (void)
Description	Makes initial port settings for the LEDs and switches.
Arguments	None
Return Value	None

irq_init	
Outline	IRQ initial settings
Header	None
Declaration	void irq_init(void)
Description	Makes initial IRQ settings.
Arguments	None
Return Value	None

Excep_IRQm	
Outline	IRQm interrupt handler
Header	None
Declaration	static void Excep_IRQm(void)
Description	LED0 is turned on in the IRQm interrupt handler. After this, the processing loops until the IRQm.IR flag is cleared (the switch is released), and LED0 is turned off.
Arguments	None
Return Value	None

Excep_IRQn	
Outline	IRQn interrupt handler
Header	None
Declaration	static void Excep_IRQn(void)
Description	LED1 is turned on in the IRQn interrupt handler. After this, the processing loops until the IRQn.IR flag is cleared (the switch is released), and LED1 is turned off.
Arguments	None
Return Value	None

5.7 Flowcharts

5.7.1 Main Processing

Figure 5.2 is a flowchart of the main processing routine.

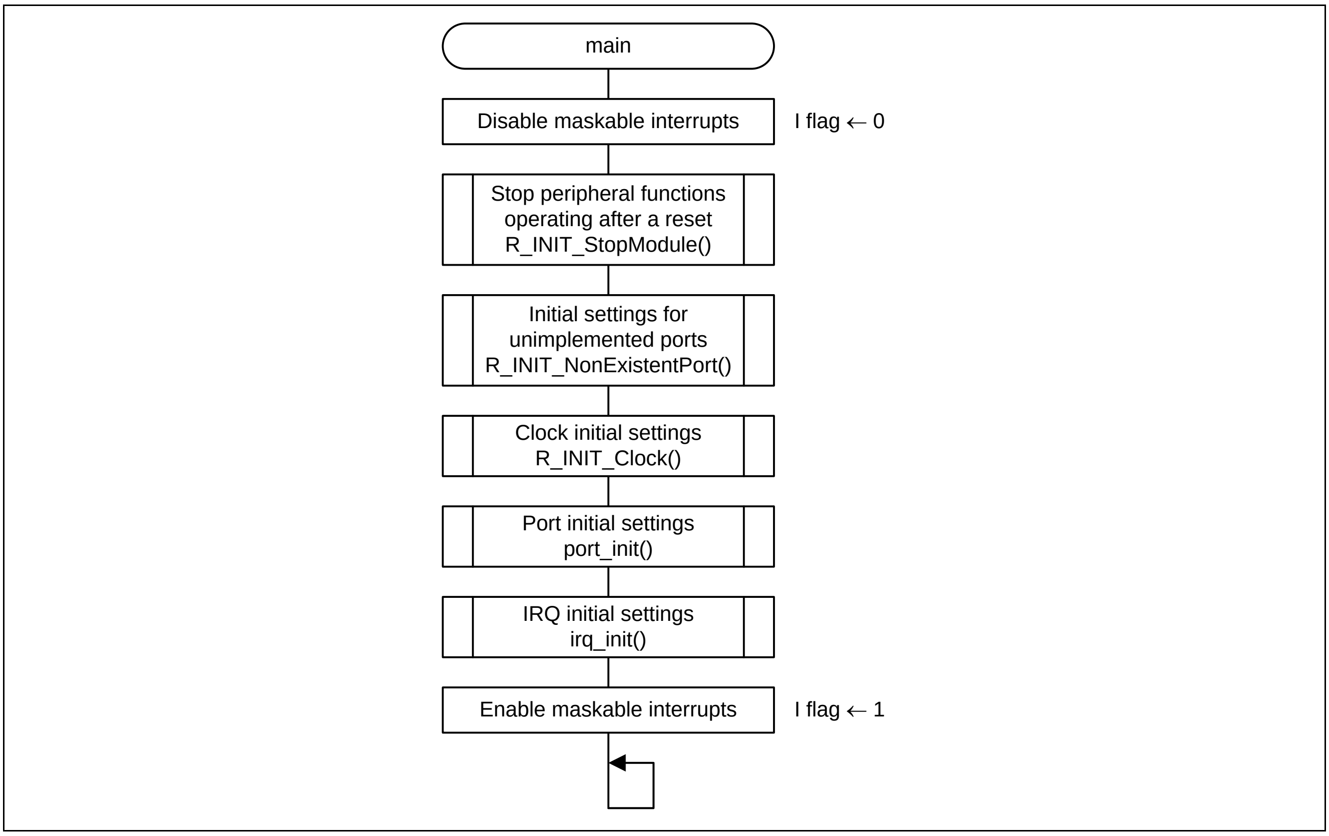
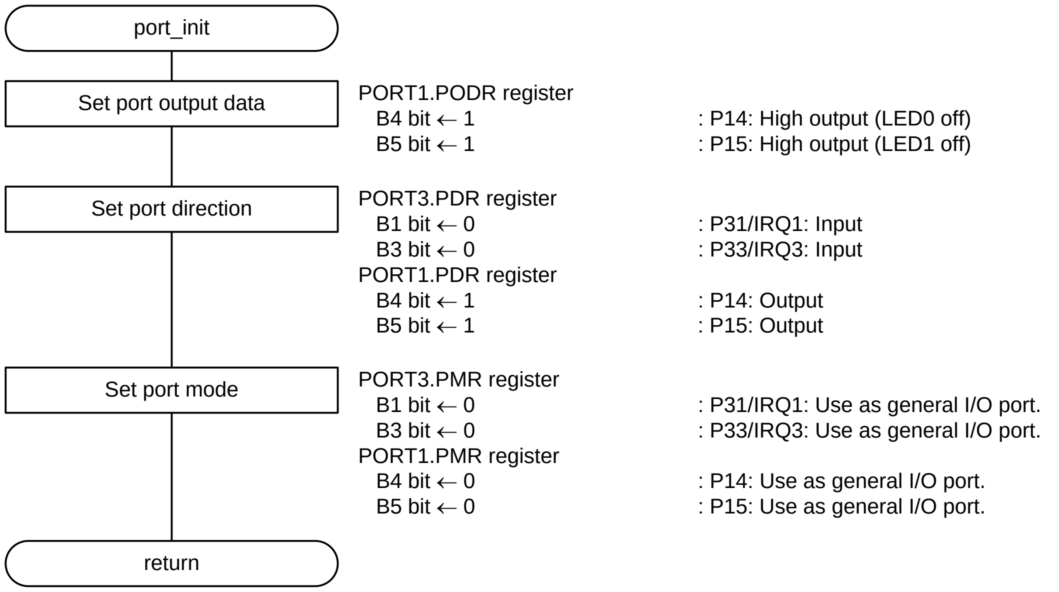


Figure 5.2 Main Processing

5.7.2 Port Initial Settings

Figure 5.3 is a flowchart of port initial settings.

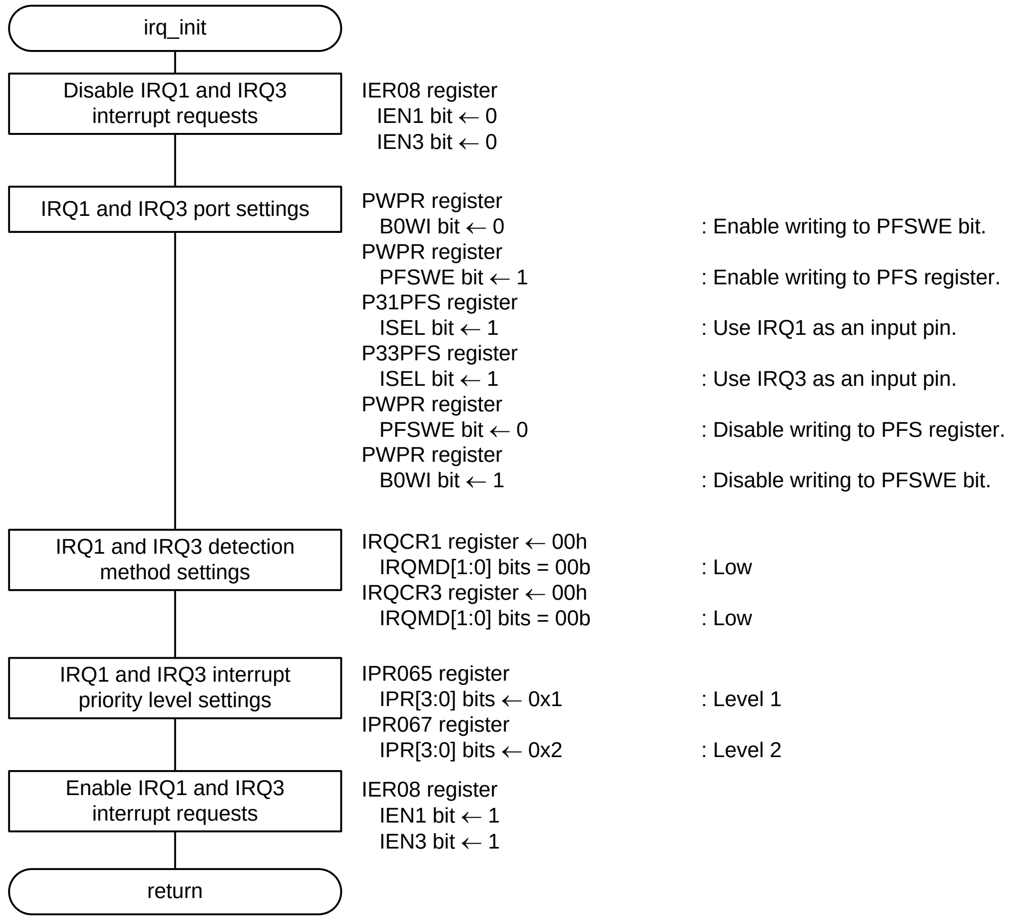


Note: The port settings used in the sample code for this application note are: LED0 = P14, LED1 = P15, IRQm = P31/IRQ1, and IRQn = P33/IRQ3.

Figure 5.3 Port Initial Settings

5.7.3 IRQ Initial Settings

Figure 5.4 is a flowchart of IRQ initial settings.



Note: The port settings used in the sample code for this application note are: IRQm = P31/IRQ1 and IRQn = P33/IRQ3.

Figure 5.4 IRQ Initial Settings

5.7.4 IRQm Interrupt Handler

Figure 5.5 is a flowchart of the IRQm interrupt handler.

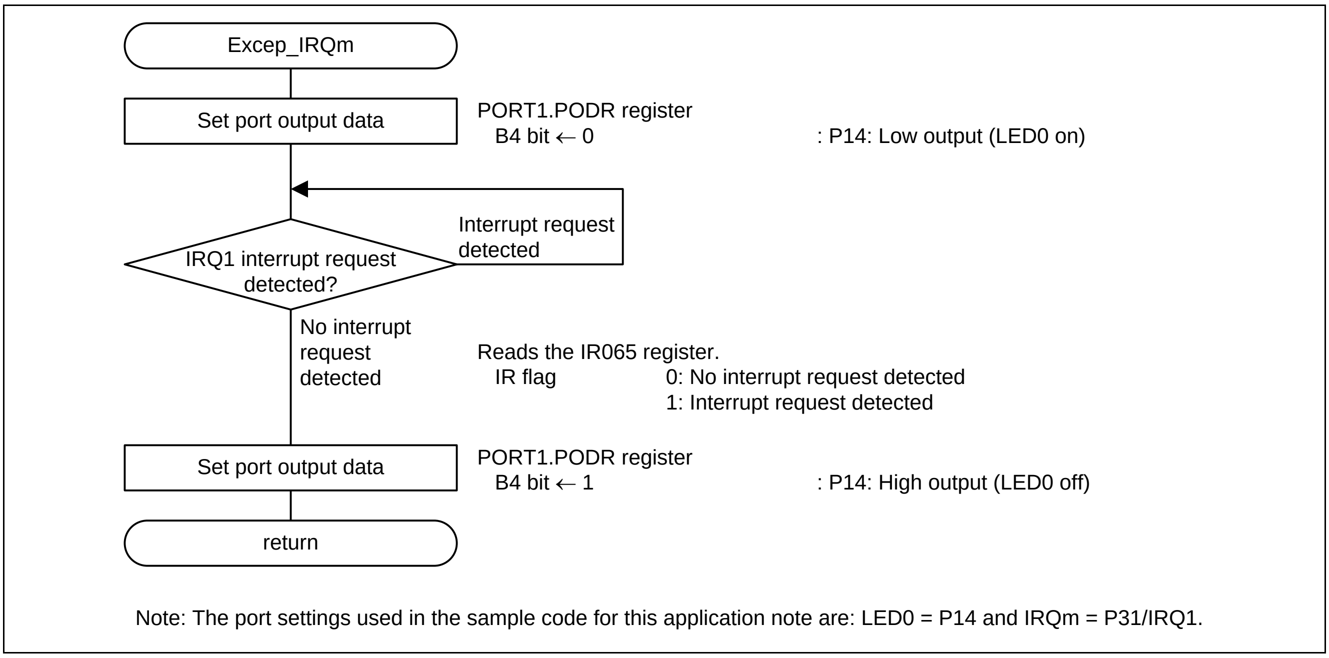


Figure 5.5 IRQm Interrupt Handler

5.7.5 IRQn Interrupt Handler

Figure 5.6 is a flowchart of the IRQn interrupt handler.

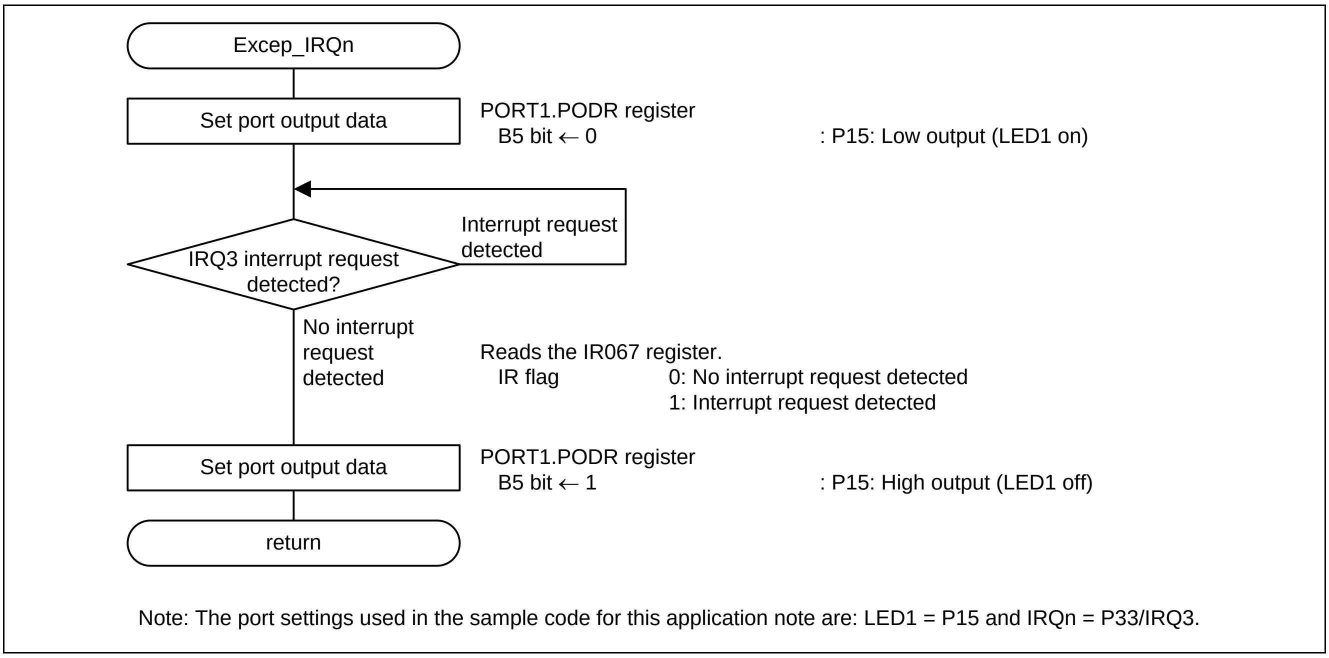


Figure 5.6 IRQn Interrupt Handler

6. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

7. Reference Documents

User's Manual: Hardware

RX220 Group User's Manual: Hardware Rev.1.00 (R01UH0292EJ)
(The latest version can be downloaded from the Renesas Electronics Web site.)

Technical Updates/Technical News

(The latest information can be downloaded from the Renesas Electronics Web site.)

User's Manual: Development Environment

RX Family C/C++ Compiler Package V.1.01 User's Manual Rev.1.00 (R20UT0570EJ)
(The latest version can be downloaded from the Renesas Electronics Web site.)

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jul 01, 2014	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to a product with a different type number, confirm that the change will not lead to problems.
   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Ku, Seoul, 135-920, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 4.0