RX600 Series

Migrating from the RX610/RX62N to the RX63N

R01AN1072EU0100
Rev. 1.00
April 07, 2012

Introduction

This document provides tips and guidelines for users porting code from the RX610/RX62x group of MCUs to the RX63x group of MCUs. It discusses the basic changes necessary to get code written for an RX610/RX62x MCU to run on an RX63x MCU. It does not, however, discuss differences in the peripherals of specific chips except in those cases where a peripheral affects the configuration of the chip (i.e. the Multi-Function Pin Controller).

Target Device

RX630, RX63N

Contents

1. Overview of Migration Tasks

While the RX family of microcontrollers shares a common core, there are some significant differences between the RX610/RX62x groups and the RX63x group. These differences are a result of improvements to increase performance or decrease power consumption of the RX63x group. Those migrating designs from the RX610 and RX62x groups to the RX63x group will need to make some simple changes to their hardware and software to get their system running properly on the RX63x group parts. The changes fall into the following categories:

1. Boot mode & chip configuration
2. Clock chain settings
3. Option-setting memory
4. Register protection
5. Multi-Function Pin Controller (MPC)
6. GPIO Pin Setup
7. Interrupt Control Unit

Each of these categories is discussed in detail in the following sections.

1.1 References

RX610 Group Hardware Manual Rev. 1.10 (R01UH0032EJ0110)

RX62N, RX621 Group Hardware Manual Rev. 1.30 (R01UH0033EJ0130)

RX63N, RX631 Group Hardware Manual Rev 1.00 (R01UH0041EJ0100)

RX630 Group Hardware Manual Rev 1.12 (R01UH0040EJ0112)

1.2 Chapters you should read first in the RX63x Hardware Manual

While most of the peripherals in the RX63x remain unchanged from the RX610/RX62x, the following chapters contain significant changes and features that must be understood before migrating designs to the RX63x. These chapters should be thoroughly reviewed (document number R01UH0041EJ0100, “RX63N, RX631 Hardware Manual”):

Chapter 7, “Option-Setting Memory” explains the new flash-based options registers that can be used to turn on such features as watchdog timers automatically at boot-up without any code intervention. It is important to understand where in memory these registers are located and what values to place into them so that unexpected behavior is avoided.

Chapter 9, “Clock Generation Circuit” details the clock chain on the RX63x which is significantly enhanced from the RX610/RX62N. The default behavior of the clock chain is very different from the RX610/RX62x, so this chapter is critical to understanding of how to get the part running at high speed after reset.

Chapter 13, “Register Write Protection Function” describes new protection features that make it impossible to change certain registers in the MCU without first unlocking the protection mechanism. Readers must understand this mechanism or they will not be able to enable peripherals or change the clock settings.

Chapter 15, “Interrupt Controller (ICUb)” illustrates changes to the interrupt controller on the RX63x. Vector numbers have changed, and in some cases interrupt sources that were previously discrete have been aggregated into shared vectors. If your application uses interrupts this chapter should be carefully reviewed.

Chapter 21, “I/O Ports” lists the registers used to control I/O pins on the RX63x. Many of these register names have changed, so code must be modified to run on the RX63x.

Chapter 22, “Multi-Function Pin Controller (MPC)” details the new way that the RX63x pins are setup and assigned to peripheral functions. You must read this code if you are using any pins for peripheral functions such as communications channels, timers, analog input/output, and interrupt lines.

Chapter 46, “ROM (Flash Memory for Code Storage)” has important information on protecting your program in memory and on using special memory regions such as the user boot area. It also contains details on using the special USB and serial boot modes.

1.3 How to Use This Document

In this document, the term “RX63x” is used to refer to members of the RX63N, RX631, and RX630 groups of microcontrollers. Similarly, the term “RX62x” is used to refer to the RX62N and RX621 groups.

2. Boot Configuration

While strapping pins are still used on the RX63x to configure the run mode of the part, new flash-based options have been added that allow automatic startup and configuration of some on-board peripherals. A single MD pin configures the RX63x for either single-chip mode or one of two special boot modes (serial boot or user/USB boot mode). Refer to chapter 3, “Operating Modes” in the RX63x Hardware Manual for details on setting the boot mode.

A new feature on the RX63x is the Option-Setting Memory. These are flash-based registers that are set when the device is programmed, and that govern the operation of the chip immediately after reset. The registers are detail in Chapter 7 of the Hardware Manual: “Option-Setting Memory”.

The flash option registers occupy space in the normal memory map. Although the registers are located in a portion of the flash memory that was reserved on the RX62x, it is possible that some customers may have stored data in these locations inadvertently. The user must check to ensure that no unwanted data is written to these locations or else unexpected behavior of the chip may result. For instance, settings in the flash option registers can enable the Independent Watchdog Timer (IWDT) immediately after reset. If data stored in program ROM inadvertently overlaps the flash option register, it is possible to turn on the IWDT on without realizing it. This will cause the debugger to have communications problems with the board.

2.1 Summary of Migration Steps for Boot Configuration

1. Check strapping of MD pin and PC7 (chapter 3, “Operating Modes”)
2. Ensure flash option registers for single-chip mode are properly initialized in code (OFS1, OFS2, MDES). See chapter 7, “Option-Setting Memory”
3. If using user boot area, flash option registers for user boot mode must be properly initialized (UB code A, UB code B, MDEB). Refer to chapter 7, “Option-Setting Memory” and chapter 46, “ROM (Flash Memory for Code Storage)”.

3. Clock Chain Differences

The clock chain in the RX63x adds a number of new features. In addition to the main clock oscillator, the sub-clock oscillator, and the independent watchdog timer oscillator from the RX610/RX62x, the RX63x adds two new on-chip oscillators: the high-speed clock oscillator (HOCO) and the low-speed clock oscillator (LOCO). Refer to the “Clock Generation Circuit” chapter of the Hardware Manual for full details.

3.1 Clock Settings after Reset

On the RX62x, the main clock oscillator is set as the default system clock with a PLL multiplier of 1x, which means that the RX62x will run at the speed of the external crystal after reset. On the RX63x, the 125 kHz low-speed clock oscillator (LOCO) is the default system clock after reset. Because of this, changes should be made to your startup code to set the system clock of the RX63x to one of the higher speed sources as soon as possible after reset. The sample code in the workspace associated with this application note demonstrates the suggested sequence.

3.2 New and Changed Clock Domains

The RX63x group has three additional clock domains. Independent clock domains are provided for the flash clock (FCLK), the Ethernet controller and Ethernet DMAC (PCLKA), the IEBus (IECLK). On the RX62x, all clock domains with the exception of the RTC & independent watchdog were driven off the output of the PLL. A new clock selector in the RX63x controlled by the CKSEL bits in register SCKR3 allows selection of five different sources for these clock domains: LOCO, HOCO, main clock oscillator, sub-clock oscillator, or the output of the PLL.

The new flash clock (FCLK) domain allows the user to adjust access time to the registers of the flash control unit (FCU). It also controls access time to the data flash. Note that FCLK does not affect read access time of the on-chip program flash for program execution; this is still 10ns. The speed of FCLK can be slowed to reduce power consumption or increased to maximize performance.

The Ethernet controller (ETHERC) and its associated Ethernet DMAC (EDMAC) are clocked by peripheral clock A (PCLKA) on the RX63x. In applications using Ethernet, PCLKA must be set equal to the main system clock (ICLK) and must be greater than 12.5 MHz.

The RX63x includes an Inter Equipment Bus (IEBus) peripheral that is clocked by the IEBus Clock (IECLK). A selector in the SCKCR2 register allows flexible configuration of the base clock for the IEBus.

The USB clock (UCLK) on the RX63x can use one of two divisors: 1/3 and 1/4; on the RX62x there is a fixed 1/4 divisor. The UCLK divisor on the RX63x is controlled by the SCKCR2 register.

3.3 Summary of Migration Steps for the Clock Chain

1. Modify startup code to configure chip for high speed operation as soon as possible after reset
2. Ensure that new & changed clock domains (PCLKA, FCLK, IECLK, UCLK) are properly configured
3. Note that the clock control registers are now protected (see section below on Register Write Protection)

4. Register Write Protection

A new safety feature on the RX63x is Register Write Protection. This feature prevents inadvertent changes to registers that are critical to the operation of the RX63x. Refer to chapter 13, “Register Write Protection Function” for details. The protection function covers three different groups of registers:

1. registers related to the clock generation circuit
2. registers related to operating modes, low power modes (including Module Stop Control), and software reset
3. registers related to low-voltage detection and battery backup

A single Protect Register (PRCR) contains bits that control write access to each of the groups listed above. A key field in the PRCR register must be unlocked before the control bits can be modified.

If your ported code is attempting to change registers and the registers are not updating as you expect, check the “Register Write Protection Function” chapter of the Hardware Manual to see if those registers are protected by the PRCR.

4.1 Summary of Migration Steps for Register Write Protection

1. Wherever existing code makes changes to one of the registers listed as protected in chapter 13, “Register Write Protection Function”, add code to unprotect those registers using PRCR. Add code to protect the registers again with PRCR after the code that modifies the registers.
2. Optionally, you may choose to disable write protection once after boot up and leave it disabled. To do this, set all of the PRCx bits in the PRCR register to a one. This will minimize code changes to the rest of your program, although it will do so at the risk of reduced security for those registers left unlocked.

5. Multi-Function Pin Controller

The Multi-Function Pin Controller (MPC) is a new feature on the RX63x that replaces the Port Function Control registers on the RX62x. The result is a much more flexible assignment of pins to peripherals functions, with a much finer granularity of selection. In summary, each pin on the part now has its own Pin Function Control Register (PFS) that defines the function for that pin. A 5-bit field in the PFS determines which peripheral function, if any, is assigned to the pin. Additional bits configure the pin for analog use or for use as a hardware interrupt line.

Note that the PFS registers in the MPC are protected by the MPC Write-Protect Register (PWPR). This register must be unlocked to allow writes to other registers in the MPC. Details on configuring the MPC can be found in chapter 22, “Multi-Function Pin Controller (MPC)”.

5.1 Summary of Migration Steps for Multi-Function Pin Controller

1. For any peripherals that are used that are tied to port pins (communications channels, IRQ pins, timer input/output lines), add code to configure the Pin Function Control Register (PFS) for those pins. Be sure to follow the sequence of operations in the "Usage Notes" section of the MPC chapter of the HW Manual.
2. For pins used for analog input, set the ASEL bit in the pin's PFS
3. For pins used as hardware IRQ signals, set the IEL bit in the PFS
4. Remember that the PFS registers are write protected by the MPC PWPR register
5. If you are using the external bus, the Ethernet controller, or USB, there are additional registers in the MPC that must be configured before using these peripherals
6. Don't forget the new Port Mode Register (PMR) in the GPIO register block (see next section for details)

6. GPIO Pin Setup

The names of the registers that control the port pins on the RX63x have changed from RX62x. In some cases the functionality has changed slightly. Refer to chapter 21, "I/O Ports" in the Hardware Manual for full details.

The following table summarizes the changes:

RX610/RX62x Register	RX63x Register	Description
PORTn.DDR	PORTn.PDR	Sets the direction of a pins (input/output)
PORTn.DR	PORTn.PODR	Changes the output level of a pin (high or low)
PORTn.PORT	PORTn.PIDR	Reads the input level of a pin
PORTn.ODR	PORTn.ODR0/1	Sets the open-drain configuration of a port
PORTn.PCR	PORTn.PCR PORTn.DSCR	Pull-up control and drive strength (drive strength available on RX63x only)
PORTn.ICR	PORTn.PMR	The ICR on the RX62x is controls the input buffer; it's necessary to set this for pins used by peripheral functions or as GPIO inputs. On the RX63x this function is replaced by the Port Mode Register (PMR) which is used to assign a port pin to a peripheral.
PFnXXX	MPC.PnxPFS	The RX62x uses Port Function Control registers to select whether peripheral functions are assigned to primary or secondary pins. The RX63x uses the MPC Pin Function Control registers (PFS) to assign peripheral functions to individual pins.

6.1 Summary of Migration Steps for GPIO

1. Review the table above and the hardware manual
2. Make changes to GPIO register names and functions
3. Don't forget to add new MPC code (see previous section)

7. Interrupt Control Unit Changes

The basic function of the Interrupt Control Unit remains unchanged; however, the version used in the RX63x (ICUb) includes three new features: Interrupt Request Groups, Interrupt Unit Selection, and Digital Filtering. These new features are discussed briefly in the following sections. Chapter 15, "Interrupt Controller (ICUb)" in the Hardware Manual has complete details.

7.1 Interrupt Request Groups

Interrupt Request Groups aggregate a number of previously separate interrupt vectors into a single vector. The sources that are grouped together are generally related. For instance, the error interrupts for all of the Serial Communications Interface (SCI) channels are grouped into a single vector. A single Interrupt Service Routine (ISR) must now sort and service all pending requests in the group.

There are 8 interrupt groups on the RX63x. Groups 0 through 6 are for edge sensitive sources such as CAN, MTU channels, and TPU channels. Group 12 is for level sensitive sources, specifically, the error interrupts from the SCI and RSPI channels.

Three extra registers in the ICU control each group: the Group Interrupt Source register (GRPm), the Group Interrupt Enable register (GENm), and the Group Interrupt Clear register (GCRm). When one of the sources feeding into the interrupt group becomes active, a corresponding bit in the GRPm register becomes set indicating a pending interrupt. If that source is enabled by the corresponding bit in the GENm register, an interrupt request is routed to the ICU through the shared vector. The ISR must then service all interrupts that are pending as indicated by the bits set in the GRPm register, clearing each as it is serviced by writing to the GCRm register.

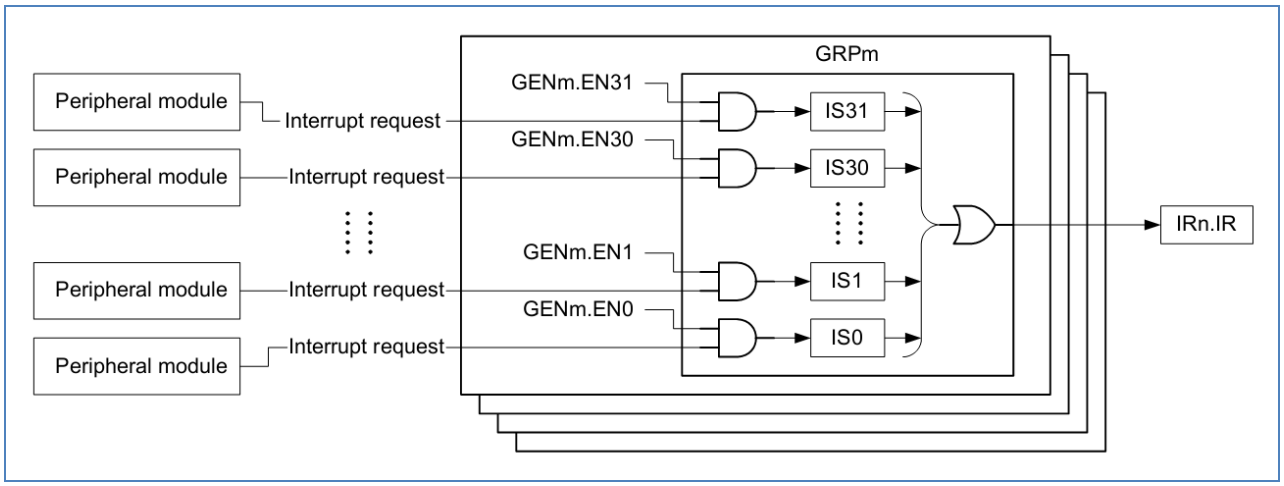

Figure 1 - Interrupt Request Groups

7.2 Interrupt Unit Selection

Interrupt Unit Selection overlays the interrupts of two different timer peripherals (MTU & TPU) onto the same set of vectors. Only one of the two peripherals can be selected to generate these interrupts. The example show below shows Interrupt Unit 0. It switches vectors 142 through 147 between sources from either MTU0 or TPU6. Refer to the Hardware Manual for the specific sources in each group.

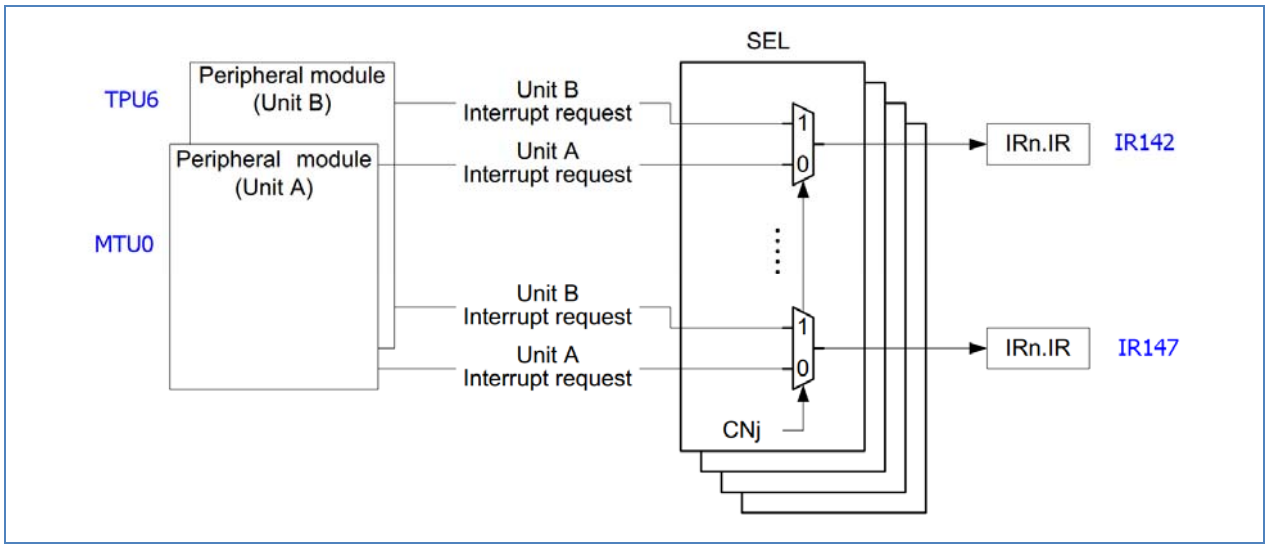

Figure 2 - Interrupt Unit Selection

7.3 IRQ Pin Digital Filtering

Hardware IRQ pins now offer digital filtering. The filters are based on repetitive sampling of the signal at one of four selectable clock rates (PCLK, PCLK/8, PCLK/32, PCLK/64). They filter out short pulses: any high or low pulse less than 3 samples at the filter rate. The filters are useful for filtering out ringing and noise in these lines, but are much too quick for filtering out long events like mechanical switch bounce. Enabling filtering adds a short bit of latency (the filter time) to the hardware IRQ lines.

7.4 Summary of Migration Steps for Interrupt Control Unit

1. Check for changes to vector numbers & update ISR prototypes as necessary
2. If any interrupts used by the application now fall into an Interrupt Group (i.e. SCI or RSPI error interrupts) or an Interrupt Unit (some channels of MTU and TPU), make changes as needed
3. Enable digital filtering if desired

8. Sample Workspace

The sample workspace that accompanies this application note includes the following features (the corresponding file/function are shown for each):

1. Updated boot code that provides early switch to high-speed operation (resetprg.c/PowerON_Reset_PC)
2. Sample code to show the use of flash-based option registers (flash_options.c)
3. Updated clock settings (resetprg.c/operating_frequency_set)
4. Use of new register protection feature (resetprg.c/operating_frequency_set)
5. Setup of the Multi-Function Pin Controller (hwsetup.c/output_ports_configure) (switches.c/switches_initialize)
6. GPIO pin setup (hwsetup.c/output_ports_configure)
7. Interrupt control unit (switches.c/switches_initialize)

Website and Support

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

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

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Apr.07.2012	—	First edition issued