RX231 Group
Microcontroller Migration Guide for Migration from H8/3048 (H8/300H Series) to RX231

Introduction
This application note describes precautions to observe when replacing the H8/3048 device with the RX231 device and provides other information such as differences between the H8/3048 and RX231 devices. For details on individual functions, refer to the latest hardware manual.

Target Device
RX231
## Contents

1. CPU Architecture ............................................................................................................. 5
   1.1 System Registers ........................................................................................................ 5
   1.1.1 General-Purpose Registers .................................................................................. 5
   1.1.2 Control Registers ................................................................................................. 6
   1.2 Option-Setting Memory ............................................................................................. 9
       1.2.1 Outline of Option-Setting Memory .................................................................... 9
       1.2.2 Endian Settings ............................................................................................... 10
   1.3 Reset Function ........................................................................................................... 11
       1.3.1 Reset Sources .................................................................................................... 11
       1.3.2 Reset Sources and Initialization Scope ............................................................. 12
   1.4 Clock Settings ............................................................................................................ 13
       1.4.1 Clock Sources .................................................................................................... 13
       1.4.2 RX231 Clock Generation Circuit ....................................................................... 14
   1.5 Operating Modes ........................................................................................................ 15
       1.5.1 Comparison of Operating Modes ....................................................................... 15
       1.5.2 Comparison of the Memory Map Between the H8/3048 and RX231 Devices ..... 16
       1.5.3 Operating Mode Settings .................................................................................. 19
   1.6 Processor Modes ......................................................................................................... 20
   1.7 Exception Handling .................................................................................................... 21
       1.7.1 Comparison of Exception Sources ...................................................................... 21
       1.7.2 Priority of Exception Sources in Exception Handling ................................. 21
       1.7.3 Basic Processing Sequence of Exception Handling ....................................... 22
       1.7.4 Vector Configuration ....................................................................................... 23
       1.7.5 Interrupt Masking by CCR (PSW) ................................................................. 24
   1.8 Interrupt Handling ....................................................................................................... 25
       1.8.1 Interrupt Controller ......................................................................................... 25
       1.8.2 Management of Interrupt Flags ....................................................................... 27
       1.8.3 Control of Fast Interrupts ................................................................................. 29
       1.8.4 Digital Filter ..................................................................................................... 30
       1.8.5 Multiple Interrupts ........................................................................................... 31

2. On-Chip Functions (Peripheral Modules) ................................................................. 32
   2.1 List of On-Chip Functions ......................................................................................... 32
   2.2 I/O Ports .................................................................................................................. 33
       2.2.1 Comparison of I/O Port Specifications ............................................................... 33
       2.2.2 I/O Settings ....................................................................................................... 34
   2.3 Buses ........................................................................................................................ 39
       2.3.1 Comparison of Specifications ........................................................................... 39
       2.3.2 Bus Configuration ............................................................................................. 40
       2.3.3 Register Configuration ....................................................................................... 42
2.3.4 Wait Settings ................................................................. 42
2.3.5 External Bus Pin Setting Example ................................................................. 43
2.4 Interrupt Controller ........................................................................ 44
2.4.1 Usage Example for IRQ Pins ................................................................. 44
2.5 DMA Controller ...................................................................... 45
2.5.1 Comparison of Specifications ................................................................. 45
2.5.2 DMAC Block Diagram ...................................................................... 47
2.5.3 List of Registers ........................................................................ 49
2.5.4 DMAC Activation Sources ...................................................................... 50
2.5.5 Transfer Modes ...................................................................... 51
2.5.6 Module Stop ...................................................................... 53
2.6 Serial Communications Interface ................................................................. 54
2.6.1 Comparison of Specifications ................................................................. 54
2.6.2 SCI-related Settings Required During Migration ........................................ 57
2.6.3 Module Stop ...................................................................... 57
2.7 High-Function Timers (ITU, MTU2a, and TPUa) ................................................ 58
2.7.1 Comparison of Specifications ................................................................. 58
2.7.2 Notes on Replacing the Timer ...................................................................... 59
2.7.3 List of Registers ...................................................................... 60
2.7.4 Module Stop ...................................................................... 61
2.8 Timer-based Pulse Output Function ................................................................. 62
2.8.1 Comparison of Specifications ................................................................. 62
2.9 Watchdog Timers ...................................................................... 65
2.9.1 Comparison of Specifications ................................................................. 65
2.10 A/D Converter ...................................................................... 66
2.10.1 Comparison of Specifications ................................................................. 66
2.10.2 Operating Modes ...................................................................... 67
2.10.3 List of Registers ...................................................................... 68
2.10.4 Module Stop ...................................................................... 69
2.11 D/A Converter ...................................................................... 70
2.11.1 Comparison of Specifications ................................................................. 70
2.11.2 Comparison of Registers ................................................................. 70
2.11.3 Module Stop ...................................................................... 70
2.12 Flash Memory ...................................................................... 71
2.12.1 Comparison of Specifications ................................................................. 71
2.13 Low-Power Modes and Module Stop Function ........................................ 72
2.13.1 Comparison of Low-Power Mode Specifications ........................................ 72
2.13.2 Mode Transitions ...................................................................... 74
2.13.3 Module Stop ...................................................................... 75
3. Reference Documents ...................................................................... 76
1. CPU Architecture

1.1 System Registers
The following sections show the differences between the system registers of the H8/3048 and the RX231 devices.

1.1.1 General-Purpose Registers
The H8/3048 and RX231 devices have 32-bit general-purpose registers. With the H8/3048 device, eight*1 32-bit general-purpose registers are available. With the RX231 device, sixteen 32-bit general-purpose registers are available. The register used as the stack pointer (SP) on the H8/3048 device is ER7, and that on the RX231 device is R0.

Note:  1. On the H8/3048 device, eight 32-bit ER general-purpose registers can be divided into two (E and R) groups of eight 16-bit registers, and the eight 16-bit registers of group R can further be divided into two groups of eight 8-bit registers. On the RX231 device, general-purpose registers cannot be divided.

Figure 1.1 shows the differences between the general-purpose registers of the H8/3048 and RX231 devices.

![Figure 1.1 Differences of General-Purpose Registers](image_url)
1.1.2 Control Registers

Figure 1.2 shows the differences between the control registers of the H8/3048 and RX231 devices.

![Figure 1.2 Differences of Control Registers](image)

The following outlines the RX231-specific control registers that are not present on the H8/3048 device.

- **Interrupt stack pointer (ISP) / user stack pointer (USP)**
  There are two types of stack pointers (SP): the interrupt stack pointer (ISP) and user stack pointer (USP). The stack pointer (ISP or USP) to be used can be selected by using the stack pointer specification bit (U) of the processor status word (PSW).

- **Interrupt table register (INTB)**
  This register is used to specify the start address of the interrupt vector table.

- **Backup PC (BPC) / backup PSW (BPSW)**
  Ordinary interrupts and fast interrupts are supported. For fast interrupts, the data of the program counter (PC) and processor status word (PSW) is saved to the dedicated registers (BPC and BPSW), thereby reducing the processing time needed to save the register data. Note that the BPC and BPSW registers do not support multiple interrupts.

- **Fast interrupt vector register (FINTV)**
  This register is used to specify the jump destination when a fast interrupt occurs.

- **Floating-point status word (FPSW)**
  There are floating-point status words for indicating floating-point arithmetic results.

- **Exception table register (EXTB)**
  This register is used to set the first address of the exception vector table.

- **Differences of the condition code register**
  On the RX231 device, the flags equivalent to the half-carry flag (H) and user bit (U) of the CCR control register on the H8/3048 device are unavailable. However, for the other bits of the CCR register, the corresponding bits are provided by the PSW register.
Figure 1.3 and Table 1.1 show the differences between the CCR register on the H8/3048 device and the PSW register on the RX231 device.

![Diagram showing differences between CCR and PSW registers](image.png)

Figure 1.3  Differences Between the CCR Register on the H8/3048 and the PSW Register on the RX231
Table 1.1  Differences Between the CCR Register on the H8/3048 and the PSW Register on the RX231

<table>
<thead>
<tr>
<th>CCR Bit Name</th>
<th>PSW Bit Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C (carry flag)</td>
<td>C</td>
<td>The calculation result indicated by the C bit in CCR on the H8/3048 is indicated by the C bit in PSW on the RX231. This flag indicates that a carry, borrow, or shift-out occurred in the calculation result.</td>
</tr>
<tr>
<td>Z (zero flag)</td>
<td>Z</td>
<td>The calculation result indicated by the Z bit in CCR on the H8/3048 is indicated by the Z bit in PSW on the RX231. This flag indicates that the calculation result was 0.</td>
</tr>
<tr>
<td>N (negative flag)</td>
<td>S</td>
<td>The calculation result indicated by the N bit in CCR on the H8/3048 is indicated by the S bit in PSW on the RX231. This flag indicates that the calculation result was negative.</td>
</tr>
<tr>
<td>V (overflow flag)</td>
<td>O</td>
<td>The calculation result indicated by the V bit in CCR on the H8/3048 is indicated by the O bit in PSW on the RX231. This flag indicates that an overflow occurred in the calculation result.</td>
</tr>
<tr>
<td>I (interrupt mask bit)</td>
<td>I</td>
<td>The I and UI bits in CCR on the H8/3048 correspond to the I bit in PSW on the RX231. 0: Interrupts are disabled. 1: Interrupts are enabled. This bit is used to enable reception of interrupt requests on the RX231. The initial state is 0, so it is necessary to set this bit to 1 to accept interrupts. Also, this bit is cleared to 0 when an exception is accepted, and no interrupts are accepted while its value remains 0. Note that the interrupt status flag of the interrupt controller is reset when an interrupt request occurs, regardless of the setting of this bit.</td>
</tr>
<tr>
<td>UI (user bit/interrupt mask bit)</td>
<td>—</td>
<td>This bit specifies the stack pointer used by the RX231. 0: Interrupt stack pointer (ISP) 1: User stack pointer (USP) This bit is cleared to 0 when an exception is accepted.</td>
</tr>
<tr>
<td>—</td>
<td>PM</td>
<td>This bit specifies the processor mode of the RX231. 0: Supervisor mode 1: User mode This bit is cleared to 0 when an exception is accepted.</td>
</tr>
<tr>
<td>—</td>
<td>IPL[3:0]</td>
<td>These bits specify the interrupt priority level. The RX231 supports level settings from 0 (lowest) to 15 (highest). Only interrupts with a priority level higher than this setting are accepted.</td>
</tr>
<tr>
<td>U (user bit)</td>
<td>—</td>
<td>There is no corresponding bit on the RX231.</td>
</tr>
<tr>
<td>H (half-carry flag)</td>
<td>—</td>
<td>There is no corresponding bit on the RX231.</td>
</tr>
</tbody>
</table>
1.2 Option-Setting Memory

On the RX231, it is necessary to set up the option-setting memory. Use the option-setting memory to configure the settings such as the operation that follows a CPU reset and the endian order of data. The option-setting memory is allocated in the ROM, and cannot be overwritten by a software program. When programming the ROM, it is necessary to also set up the option-setting memory.

1.2.1 Outline of Option-Setting Memory

Figure 1.4 shows the areas in the option-setting memory.

An outline of the registers is provided below:

- **Endian select register (MDE)**
  Register for configuring the CPU endian settings

- **Optional function select register 1 (OFS1)**
  The OFS1 register is used to configure the following settings:
  - Whether to enable or disable the function that resets voltage monitor 0 after a reset, and the level settings for voltage detector 0
  - Whether to enable or disable the function that activates the high-speed on-chip oscillator (HOCO) after a reset
  - Time before startup when power is turned on (normal/short)

- **Optional function select register 0 (OFS0)**
  The OFS0 register is used to determine how the independent watchdog timer (IWDT) and watchdog timer (WDT) will operate after a reset.

Figure 1.5 shows sample option-setting memory settings.

```
/^ Settings for single-chip mode and big-endian */
#pragma section C EXCEPTVECT
void (*const Except_Vectors[])(void) = {
  //:0xfffffff8 MDE register
  #ifdef __BIG
    (void (*)(void))0xfffffff8, // big
  #else
    (void (*)(void))0xffffffff, // little
  #endif
}
```

Figure 1.5 Example of Endian Settings
Figure 1.6 shows an example of the settings for OFS0 and OFS1.

```
#pragma section C.EXCEPTVECT
void (*const Except_Vectors[])(void) = {
    //;0xffffff88 OFS1 register
    (void (*)(void))0xffffffff, // OFS1
    //;0xffffff8c OFS0 register
    (void (*)(void))0xffffffff, // OFS0
```

Figure 1.6 Example of the Settings for OFS0 and OFS1

### 1.2.2 Endian Settings

The H8/3048 device supports only big-endian order. On the RX231 device, only little-endian is supported for instructions, and little-endian or big-endian can be selected for data. The endian settings are specified by means of the endian select bits (MDE[2:0]) of the MDE register in the option-setting memory.

If the big-endian order is to be used on the RX231 device after migration from the H8/3048 device, big-endian can be specified in the option settings of the genuine Renesas compiler. This allows migration without the need to consider endianness in the user program.

The endian settings can be switched for each CS area in the external address space. However, instruction code cannot be allocated to an external space with endian settings that differ from those of the chip. When allocating instruction code to an external space, allocate the code to an area with the same endian settings as the chip. (For details, refer to the "RX230 Group, RX231 Group User’s Manual: Hardware".)

In actuality, code such as that shown in Figure 1.5 is generated automatically according to the compiler option settings. Figure 1.7 shows an overview of specifying the endian settings by using compiler options.

![Diagram showing the process of specifying endian settings](image)

Figure 1.7 Specifying Endian Settings by Using Compiler Options
1.3 Reset Function
This section describes the differences in reset types between the H8/3048 and RX231 devices.

1.3.1 Reset Sources
Table 1.2 lists the reset sources of the H8/3048 and RX231 devices.

Table 1.2 Reset Sources

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>RES# pin reset</td>
</tr>
<tr>
<td></td>
<td>Power-on reset</td>
</tr>
<tr>
<td></td>
<td>Voltage monitor 0 reset</td>
</tr>
<tr>
<td></td>
<td>Voltage monitor 1 reset</td>
</tr>
<tr>
<td></td>
<td>Voltage monitor 2 reset</td>
</tr>
<tr>
<td></td>
<td>Independent watchdog timer reset</td>
</tr>
<tr>
<td></td>
<td>Watchdog timer reset</td>
</tr>
<tr>
<td></td>
<td>Software reset</td>
</tr>
</tbody>
</table>

(1) Reset vector configuration
The reset vector of the H8/3048 device is assigned to vector number 0 (vector address: H’0000 to H’0003).

The RX231 device has a single reset vector for multiple reset sources. The reset source is identified by using reset status registers 0 to 2 during reset processing, and the processing appropriate for the identified source is performed.
### 1.3.2 Reset Sources and Initialization Scope

On the H8/3048 device, a reset caused by the RES# pin initializes the CPU and all the internal peripheral modules. On the RX231 device, there is a flag related to reset detection for each reset source. After recovery from a reset, the reset source can be identified by checking the flags related to reset detection. Table 1.3 lists the reset types and their initialization scope on the RX231.

<table>
<thead>
<tr>
<th>Table 1.3</th>
<th>Reset Initialization Scope for the RX231 Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset Target</td>
<td>RES# pin reset</td>
</tr>
<tr>
<td>Power-on reset detection flag</td>
<td>○</td>
</tr>
<tr>
<td>Cold start/warm start determination flag</td>
<td>—</td>
</tr>
<tr>
<td>Voltage monitor 0 reset detection flag/ battery back-up function registers</td>
<td>○</td>
</tr>
<tr>
<td>Independent watchdog timer reset detection flag/independent watchdog timer registers</td>
<td>○</td>
</tr>
<tr>
<td>Watchdog timer reset detection flag/watchdog timer registers</td>
<td>○</td>
</tr>
<tr>
<td>Voltage monitor 1 reset detection flag/voltage monitor function 1 registers</td>
<td>○</td>
</tr>
<tr>
<td>Voltage monitor 2 reset detection flag/voltage monitor function 2 registers</td>
<td>○</td>
</tr>
<tr>
<td>Software reset detection flag</td>
<td>○</td>
</tr>
<tr>
<td>Realtime clock registers</td>
<td>—</td>
</tr>
<tr>
<td>Registers other than the above, CPU, and internal state</td>
<td>○</td>
</tr>
</tbody>
</table>

○: Initialization occurs.
—: No change occurs.

Notes:
1. Initialization occurs when power is turned on.
2. Some control bits are initialized by any types of resets.
1.4 Clock Settings

1.4.1 Clock Sources

This section describes the types of clock sources. On the H8/3048 device, clocks are not controlled by software. On an RX device, however, each clock must be configured. Therefore, the clock source and clock frequency can be switched. Table 1.4 lists the clock sources of the H8/3048 and RX231 devices.

Table 1.4 List of Clock Sources of the H8/3048 and RX231 Devices

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>• System clock</td>
<td>• Main clock oscillator</td>
</tr>
<tr>
<td></td>
<td>• Subclock oscillator</td>
</tr>
<tr>
<td></td>
<td>• PLL circuit</td>
</tr>
<tr>
<td></td>
<td>• USB-dedicated PLL circuit</td>
</tr>
<tr>
<td></td>
<td>• High-speed on-chip oscillator (HOCO)</td>
</tr>
<tr>
<td></td>
<td>• Low-speed on-chip oscillator (LOCO)</td>
</tr>
<tr>
<td></td>
<td>• IWDT-dedicated on-chip oscillator</td>
</tr>
</tbody>
</table>
1.4.2 RX231 Clock Generation Circuit

On the RX231 device, the LOCO is selected as the system clock after a reset. During system initialization, oscillate necessary ones of the clock sources shown in Figure 1.8 to determine the system clock. When switching the selection of the clock oscillators and system clocks, take into account the oscillation sequence and the clock oscillation stabilization time.

For details on the setting procedure, refer to the following application note:
- RX231 Group Initial Setting (R01AN2185EJ)

Figure 1.8 shows the clock generation circuit of the RX231 device.
1.5 Operating Modes

1.5.1 Comparison of Operating Modes

On the H8/3048 device, mode settings are configured by means of pins alone. On the RX231 device, however, system registers must also be configured in addition to pin configuration. For details on operating mode settings, refer to section 1.5.3, “Operating Mode Settings”, and the “RX230 Group, RX231 Group User’s Manual: Hardware”. Table 1.5 lists the operating modes of the RX231 device.

Table 1.5 Operating Modes of the RX231 Device

<table>
<thead>
<tr>
<th>Operating Modes of the RX231 Device</th>
<th>Mode Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-chip mode*1</td>
<td>Operating mode in which the on-chip ROM is enabled and the external address space is disabled</td>
</tr>
<tr>
<td>On-chip ROM enabled extended mode*1</td>
<td>Operating mode in which the on-chip ROM is enabled and the external address space is enabled</td>
</tr>
<tr>
<td>On-chip ROM disabled extended mode</td>
<td>Operating mode in which the on-chip ROM is disabled and the external address space is enabled</td>
</tr>
<tr>
<td>Boot mode</td>
<td>Operating mode in which the on-chip flash memory rewrite program (boot program), which is stored in a dedicated area inside the microcontroller, runs. The on-chip ROM can be programmed by a device outside of the microcontroller by using the USB or asynchronous serial interface (SCI1).</td>
</tr>
</tbody>
</table>

Note: 1. The on-chip ROM can be programmed without resetting the system by means of the ROM rewrite routine in the user program.
1.5.2 Comparison of the Memory Map Between the H8/3048 and RX231 Devices

Figure 1.9 compares the memory maps between the H8/3048 device in mode 6 and the RX231 device in on-chip ROM enabled extended mode.

<table>
<thead>
<tr>
<th>H8/3048: mode 6</th>
<th>RX231: on-chip ROM enabled extended mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 0000h</td>
<td>0000 0000h</td>
</tr>
<tr>
<td>02 0000h</td>
<td>0001 0000h</td>
</tr>
<tr>
<td>FF EF10h</td>
<td>0008 0000h</td>
</tr>
<tr>
<td>FF FF10h</td>
<td>0010 0000h</td>
</tr>
<tr>
<td>FF FF1Ch</td>
<td>0010 0000h</td>
</tr>
<tr>
<td>FF FFFFh</td>
<td>007F C000h</td>
</tr>
<tr>
<td></td>
<td>007F C500h</td>
</tr>
<tr>
<td></td>
<td>007F FC00h</td>
</tr>
<tr>
<td></td>
<td>0080 0000h</td>
</tr>
<tr>
<td></td>
<td>007F C000h</td>
</tr>
<tr>
<td></td>
<td>007F C500h</td>
</tr>
<tr>
<td></td>
<td>0080 0000h</td>
</tr>
<tr>
<td></td>
<td>0500 0000h</td>
</tr>
<tr>
<td></td>
<td>0800 0000h</td>
</tr>
<tr>
<td></td>
<td>FFF8 0000h</td>
</tr>
<tr>
<td></td>
<td>FFFF FFFFh</td>
</tr>
</tbody>
</table>

- **On-chip ROM**
- **External address space**
  - (areas 0 to 7)
- **On-chip RAM**
- **Internal I/O registers**
- **Reserved area**
- **Peripheral I/O registers**
- **Program ROM**
- **E2 Data Flash**
- **Read-only**
- **Reserved area**

Note: 1. ROM and RAM capacities differ, depending on the product version.

Figure 1.9 Comparison of the Memory Map Between the H8/3048 Device and the RX231 Device (in On-chip ROM Enabled Extended Mode)
Figure 1.10 compares the memory maps of the H8/3048 device in mode 3/4 and the RX231 device in on-chip ROM disabled extended mode.

![Memory Map Diagram]

**H8/3048: modes 3 and 4**

00 0000h
External address space (areas 0 to 7)

FF EF10h
On-chip RAM

FF FF1Ch
External address space

FF FFFFh
Internal I/O registers

**RX231: on-chip ROM disabled extended mode**

0000 0000h
RAM*1

0001 0000h
Reserved area

0008 0000h
Peripheral I/O registers

0010 0000h
Reserved area

0500 0000h
External address space (CS area)

0800 0000h
Reserved area

FF00 0000h
External address space

FFFF FFFFh
Internal I/O registers

**Note:** 1. ROM and RAM capacities differ, depending on the product version.

**Figure 1.10** Comparison of the Memory Map Between the H8/3048 Device and the RX231 Device (in On-chip ROM Disabled Extended Mode)
Figure 1.11 compares the memory maps of the H8/3048 device in mode 7 and the RX231 device in single-chip mode.

<table>
<thead>
<tr>
<th>H8/3048: mode 7</th>
<th>RX231: single-chip mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 0000h</td>
<td>RAM*¹</td>
</tr>
<tr>
<td>01 FFFFh</td>
<td>Reserved area</td>
</tr>
<tr>
<td>FF EF10h</td>
<td>Peripheral I/O registers</td>
</tr>
<tr>
<td>FF FF0Fh</td>
<td>Reserved area</td>
</tr>
<tr>
<td>FF FF1Ch</td>
<td>Peripheral I/O registers</td>
</tr>
<tr>
<td>FF FFFFh</td>
<td>Reserved area</td>
</tr>
<tr>
<td></td>
<td>Internal I/O registers</td>
</tr>
<tr>
<td></td>
<td>0000 0000h</td>
</tr>
<tr>
<td></td>
<td>0001 0000h</td>
</tr>
<tr>
<td></td>
<td>0010 0000h</td>
</tr>
<tr>
<td></td>
<td>0010 2000h</td>
</tr>
<tr>
<td></td>
<td>007F C000h</td>
</tr>
<tr>
<td></td>
<td>007F C500h</td>
</tr>
<tr>
<td></td>
<td>007F FC00h</td>
</tr>
<tr>
<td></td>
<td>0080 0000h</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FFF8 0000h</td>
</tr>
<tr>
<td></td>
<td>FFFF FFFFh</td>
</tr>
</tbody>
</table>

Note: 1. ROM and RAM capacities differ, depending on the product version.

Figure 1.11  Comparison of the Memory Map Between the H8/3048 Device and the RX231 Device (in Single-chip Mode)
1.5.3 Operating Mode Settings

The operating modes of the RX231 device can be categorized into two types: modes that can be selected by pin levels when a reset is canceled, and modes that can be selected by software after a reset is canceled.

Table 1.6 lists the operating modes that are determined by mode pin settings. Table 1.7 lists the operating modes that are determined by software after a reset is canceled. The software sets whether to enable or disable the on-chip ROM and whether to enable or disable the external bus based on the settings of the SYSCR0 register. For more detailed specifications, refer to the “RX230 Group, RX231 Group User’s Manual: Hardware”.

Table 1.6 Pin Settings and Operating Modes of the RX231 Device

<table>
<thead>
<tr>
<th>Pin</th>
<th>UB</th>
<th>Mode Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>—</td>
<td>Single mode</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>Boot mode (SCI interface)</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Boot mode (USB interface)</td>
</tr>
</tbody>
</table>

Table 1.7 SYSCR0 Register Settings and Operating Modes of the RX231 Device

<table>
<thead>
<tr>
<th>SYSCR0 Register</th>
<th>EXBE</th>
<th>Mode Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROME*1 0 (Disable the on-chip ROM)</td>
<td>0 (Disable the external bus)</td>
<td>Single-chip mode</td>
</tr>
<tr>
<td>1 (Enable the on-chip ROM)</td>
<td>0 (Disable the external bus)</td>
<td>On-chip ROM disabled extended mode</td>
</tr>
<tr>
<td>0 (Disable the on-chip ROM) 1 (Enable the on-chip ROM)</td>
<td>1 (Enable the external bus)</td>
<td>On-chip ROM enabled extended mode</td>
</tr>
</tbody>
</table>

Note: 1. Once the ROME bit is cleared to 0, it can no longer be set back to 1.
1.6 Processor Modes

The RX231 device supports two processing modes: supervisor mode and user mode. These two processor modes can be used to provide a hierarchical protection mechanism for CPU resources.

During migration from the H8/3048 device, using the RX231 device in only supervisor mode allows for software replacement without consideration of processor mode.

Table 1.8 Processor Modes

<table>
<thead>
<tr>
<th>Processor Modes</th>
<th>Migration Conditions</th>
<th>Brief Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervisor mode</td>
<td>• Reset cancellation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Exception occurrence</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(The PSW.PM bit is cleared to “0”.)</td>
<td>All CPU resources are accessible, and all instructions can be executed (no limitations). This is the mode in which the OS and other system programs ordinarily operate.</td>
</tr>
<tr>
<td>User mode</td>
<td>The PSW.PM bit is set to “1”.</td>
<td>Note that, in this case, the following conditions must also be satisfied: After the PSW.PM bit saved in a stack is set to 1, the RTE instruction is executed. Alternatively, after the PSW.PM bit saved in the BPSW register is set to 1, the RTFI instruction is executed. For some CPU resources, write access is restricted and privileged instructions cannot be used. These resources include the BPC register, BPSW register, and some bits in the PSW register. This is the mode in which user programs such as application programs ordinarily operate.</td>
</tr>
</tbody>
</table>

Switchover from supervisor mode to user mode:

The mode switches to user mode by using either of the following procedures:
1) Set the PSW.PM bit saved in a stack to “1”, and then execute the RTE instruction.
2) Set the PSW.PM bit saved in the backup PSW (BPSW) register to “1”, and then execute the RTFI instruction.

When the mode changes to user mode, the stack pointer specification bit (U) of the PSW register is set to “1”.

Switchover from user mode to supervisor mode:

When an exception occurs, the PSW.PM bit is set to “0” and the CPU is placed in supervisor mode. Hardware preprocessing is performed in supervisor mode. The processor mode in which the CPU was placed immediately before the exception occurs is held by the PSW.PM bit that is saved.
1.7 Exception Handling

This section describes the differences between the H8/3048 and RX231 devices in regard to handling of general exceptions (including interrupts).

1.7.1 Comparison of Exception Sources

Table 1.9 compares the exception sources in the H8/3048 and RX231 devices.

Table 1.9 Comparison of Exception Sources

<table>
<thead>
<tr>
<th>Source</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>Interrupt</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>NMI</td>
<td>Implemented</td>
<td>Implemented*1, *2</td>
</tr>
<tr>
<td>Internal</td>
<td>Implemented</td>
<td>Implemented*1, *2</td>
</tr>
<tr>
<td>External</td>
<td>Implemented</td>
<td>Implemented*1, *2</td>
</tr>
<tr>
<td>Undefined instruction exception</td>
<td>Not implemented</td>
<td>Implemented*3</td>
</tr>
<tr>
<td>Privileged instruction exception</td>
<td>Not implemented</td>
<td>Implemented*4</td>
</tr>
<tr>
<td>Access exception</td>
<td>Not implemented</td>
<td>Implemented*5</td>
</tr>
<tr>
<td>Floating-point exception</td>
<td>Not implemented</td>
<td>Implemented*6</td>
</tr>
<tr>
<td>Trap</td>
<td>Implemented (TRAPA instruction)</td>
<td>Implemented (INT or BRK instruction)</td>
</tr>
</tbody>
</table>

Notes: 1. Fast interrupts are supported. (Fast interrupts have priority level 15, the highest level.)
2. Figure 1.12 shows interrupt exception handling.
3. This exception source occurs upon detection of an attempt to execute an undefined instruction (an unimplemented instruction).
4. A privileged instruction exception occurs when a privileged instruction is used while the CPU is operating in user mode.
5. This exception source occurs upon detection of an error occurring in memory access from the CPU.
6. This exception source occurs upon detection of an unimplemented process and five exception events (overflow, underflow, abnormal accuracy, division by zero, and invalid operation).

1.7.2 Priority of Exception Sources in Exception Handling

Table 1.10 shows the priority of exception sources in the H8/3048 and RX231 devices.

Table 1.10 Priority of Exception Sources

<table>
<thead>
<tr>
<th>Priority</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Reset</td>
<td>Reset</td>
</tr>
<tr>
<td></td>
<td>External interrupt (NMI)</td>
<td>Non-maskable interrupt</td>
</tr>
<tr>
<td></td>
<td>— (Not applicable)</td>
<td>Fast interrupt</td>
</tr>
<tr>
<td></td>
<td>Interrupt (internal/external)</td>
<td>Interrupt (internal/external)*1</td>
</tr>
<tr>
<td></td>
<td>— (Not applicable)</td>
<td>Instruction access exception</td>
</tr>
<tr>
<td></td>
<td>— (Not applicable)</td>
<td>Undefined instruction exception or privileged instruction exception</td>
</tr>
<tr>
<td></td>
<td>Trap instruction</td>
<td>Unconditional trap</td>
</tr>
<tr>
<td></td>
<td>— (Not applicable)</td>
<td>Operand access exception</td>
</tr>
<tr>
<td></td>
<td>— (Not applicable)</td>
<td>Floating-point exception</td>
</tr>
</tbody>
</table>

Note: 1. The priority of these interrupts is determined by the interrupt controller.
1.7.3 Basic Processing Sequence of Exception Handling

Figure 1.12 shows the processing sequence of interrupt exception handling (internal/external) on the H8/3048 and RX231 devices.

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>User program running</td>
<td>User program running</td>
</tr>
<tr>
<td>Interrupt generation (internal/external) Generate an interrupt if the interrupt mask setting of the I bit in CCR is 0</td>
<td>Interrupt generation (internal/external) Generate an interrupt if the I bit in PSW is 1 and the interrupt priority level is higher than the processor interrupt priority level set for IPL[3:0]</td>
</tr>
<tr>
<td>Save CCR to the stack</td>
<td>Save PSW to the interrupt stack</td>
</tr>
<tr>
<td>Save PC to the stack Set PC as the address of the first instruction to be executed after control returns*¹</td>
<td>Clear the PM, U, and I bits in PSW to 0</td>
</tr>
<tr>
<td>Write 1 to the I bit in CCR</td>
<td>Save PC for the next instruction to the stack*²</td>
</tr>
<tr>
<td>Interrupt handler specified by vector setting</td>
<td>Write the interrupt priority level to IPL[3:0] in PSW</td>
</tr>
<tr>
<td>Run interrupt handler</td>
<td>Interrupt handler specified by vector setting</td>
</tr>
<tr>
<td>Restore PC and CCR from the stack</td>
<td>Run interrupt handler</td>
</tr>
<tr>
<td>Return to user program</td>
<td>Restore PC and PSW from the stack</td>
</tr>
<tr>
<td></td>
<td>Clear the LI flag</td>
</tr>
<tr>
<td></td>
<td>Return to user program</td>
</tr>
</tbody>
</table>

Notes:
1. On the H8/3048, no interrupts are accepted during execution of the LDC, ANDC, ORC, or XORC instruction, so this ensures that the next instruction is executed without fail.
2. On the RX, when the instruction being executed is RMPA, SCMPU, SMOVB, SMOVF, SMOVU, SSTR, SUNTIL, or SWHILE, the PC for that instruction is saved to the stack.

Figure 1.12 Processing Sequence of Interrupt Exception Handling (Internal/External)
1.7.4 Vector Configuration

On the H8/3048 device, only fixed vectors are used. On the RX231 device, however, the vectors are divided into two vector tables: exception vector table and interrupt vector table. On the RX231 device, the interrupt vector table is set before interrupts are enabled. The interrupt vector table is set by setting the start address of the interrupt vector table in the INTB register. Exception vectors cannot be reallocated because they are assigned to resets and other system exceptions. For information on reallocation of vector addresses, refer to the “RX230 Group, RX231 Group User’s Manual: Hardware”.

Figure 1.13 shows the configuration of the interrupt vector table in the RX231 device.

```
<table>
<thead>
<tr>
<th>b31</th>
<th>b0</th>
</tr>
</thead>
<tbody>
<tr>
<td>IntBase</td>
<td></td>
</tr>
</tbody>
</table>
```

Interrupt vectors are allocated in this order.

Figure 1.13 Configuration of the Interrupt Vector Table in the RX231 Device
1.7.5 Interrupt Masking by CCR (PSW)
On the H8/3048 device, a maximum of three interrupt mask levels can be set by using a combination of the I and UI bits in CCR when the UE bit in SYSCR is set to 0. On the RX231 device, a maximum of 16 interrupt mask levels can be set by using a combination of the IPL and I bits in PSW.

Table 1.11 lists the interrupt-related bits in PSW.

<table>
<thead>
<tr>
<th>Bits in PSW on the RX231</th>
<th>Description</th>
</tr>
</thead>
</table>
| IPL[3:0]                 | Interrupt mask level (priority level) set by MPU  
                          | Value that can be set: 0 to Fh (level 0 to 15)  
                          | When an interrupt request occurs, the priority level of the interrupt source is compared with the mask level set by these bits. If the priority level is higher than the mask level, the interrupt is accepted. |
| I                        | Value set by the I bit (interrupt enable bit):  
                          | 0: Interrupts are disabled.  
                          | 1: Interrupts are enabled.  
                          | After a system reset occurs, no interrupts are accepted unless this bit is set to “1”. This bit is cleared to “0” when an interrupt is accepted.  
                          | While the value of this bit is “0”, no interrupts are accepted. However, “1” is set for the interrupt status flag in the interrupt controller when an interrupt is generated. |
1.8 Interrupt Handling

This section compares the interrupt controller functions of the H8/3048 and RX231 devices.

1.8.1 Interrupt Controller

Table 1.12 shows the differences in the interrupt controller specifications.

**Table 1.12 Differences in the Interrupt Controller Between the Devices**

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupts</td>
<td>• Interrupts from peripheral modules</td>
<td>• Interrupts from peripheral modules</td>
</tr>
<tr>
<td></td>
<td>• Interrupt detection: Edge/level*1</td>
<td>• Interrupt detection: Edge/level*1</td>
</tr>
<tr>
<td></td>
<td><strong>External function interrupts</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• IRQ0 to IRQ5 pins</td>
<td>• IRQ0 to IRQ7 pins</td>
</tr>
<tr>
<td></td>
<td>• Sources: 6</td>
<td>• Sources: 8</td>
</tr>
<tr>
<td></td>
<td>• Interrupt detection: Low level or falling edge can be specified for each source.</td>
<td>• Interrupt detection: Low level, falling edge, rising edge, or both edges can be specified for each source.</td>
</tr>
<tr>
<td></td>
<td><strong>Software interrupts</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>—</td>
<td>Number of interrupt sources due to a write to a register: 1</td>
</tr>
<tr>
<td></td>
<td><strong>Event link interrupts</strong></td>
<td>ELSR8I, ELSR18I, and ELSR19I interrupts are generated by ELC events.</td>
</tr>
<tr>
<td></td>
<td>—</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Interrupt priority</strong></td>
<td>Priority/non-priority levels can be specified for each source or each module by configuring register settings.</td>
</tr>
<tr>
<td></td>
<td>• Priority/non-priority levels can be specified for each source or each module by configuring register settings.</td>
<td>A level from 0 to Fh can be specified for each source by configuring register settings.</td>
</tr>
<tr>
<td></td>
<td><strong>Fast interrupts</strong></td>
<td>Fast interrupts supported</td>
</tr>
<tr>
<td></td>
<td>—</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>DTC/DMAC control</strong></td>
<td>Activation supported*2</td>
</tr>
<tr>
<td></td>
<td>Activation supported*2</td>
<td>Activation supported*2</td>
</tr>
<tr>
<td></td>
<td><strong>Non-maskable interrupts</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Interrupt from the NMI pin</td>
<td>• Interrupt from the NMI pin</td>
</tr>
<tr>
<td></td>
<td>• Interrupt detection: falling edge/rising edge</td>
<td>• Interrupt detection: falling edge/rising edge</td>
</tr>
<tr>
<td></td>
<td>• Digital filter function supported</td>
<td>• Digital filter function supported</td>
</tr>
<tr>
<td></td>
<td><strong>Other sources</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Trap instruction</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Exiting the low-power mode</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Sleep mode/deep sleep mode: Non-maskable interrupt and all interrupt sources can trigger exiting the mode.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Software standby mode: Non-maskable interrupts, IRQ0 to IRQ7 interrupts, RTC alarm/ cycle interrupts can trigger exiting the mode.</td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**
1. The detection method is fixed for fixed-connection peripheral modules.
2. On the H8/3048 device, activation sources are set in the DMAC. On the RX231 device, they are set in the interrupt controller.
The interrupt controller of the H8/3048 device controls IRQ interrupt flags only, while peripheral module interrupt flags are controlled by the peripheral modules. On the RX231 device, the interrupt request registers of the interrupt controller control the interrupt flags for IRQ and all peripheral modules. The interrupt controller also controls DMAC activation source settings. On the H8/3048 device, the DMAC provides a function that disables transfer when an NMI interrupt occurs. On the RX231 device, however, no such function is available.

Table 1.13 compares the interrupt controllers of the devices.

### Table 1.13 Differences in Interrupt Controller Registers Between the Devices

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System control register (SYSCR)</strong></td>
<td><strong>Low power consumption</strong></td>
</tr>
<tr>
<td><strong>Software standby (SSBY)</strong></td>
<td><strong>Standby control register (SBYCR)</strong></td>
</tr>
<tr>
<td><strong>Standby timer select 2 to 0 (STS)</strong></td>
<td><strong>— (There is no corresponding register.)</strong></td>
</tr>
<tr>
<td><strong>User bit enable (UE)</strong></td>
<td><strong>— (There is no corresponding register.)</strong></td>
</tr>
<tr>
<td><strong>RAM enable (RAME)</strong></td>
<td><strong>Operating mode</strong></td>
</tr>
<tr>
<td><strong>NMI edge select (NMIEG)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Interrupt priority register A/B (IPRA/IPRB)</strong></td>
<td><strong>Interrupt source priority register n (IPRn)</strong></td>
</tr>
<tr>
<td><strong>Interrupt controller</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Interrupt controller</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Peripheral module</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Interrupt controller</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Peripheral module</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>DMA controller</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Data transfer control register (DTCR)</strong></td>
<td><strong>DMAC activation source select register m (DMRSRm)</strong></td>
</tr>
<tr>
<td><strong>Fast interrupt setting register (FIR)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Software interrupt generation register (SWINTR)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>DTC transfer request enable register n (DTCERn)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>IRQ pin digital filter enable register 0 (IRQFLTE0)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>IRQ pin digital filter setting register 0 (IRQFLTC0)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Non-maskable interrupt status register (NMISR)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Non-maskable interrupt enable register (NMIER)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>Non-maskable interrupt status clearing register (NMICLR)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>NMI pin digital filter enable register (NMIFLTE)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
<tr>
<td><strong>NMI pin digital filter setting register (NMIFLTC)</strong></td>
<td><strong>Interrupt controller (ICUb)</strong></td>
</tr>
</tbody>
</table>
1.8.2 Management of Interrupt Flags

When a peripheral module of the H8/3048 device generates an interrupt on detection of an edge, the corresponding interrupt flag (interrupt source flag) is cleared (the flag is cleared and a dummy read is performed) in the interrupt handler. This is done because the interrupt will be generated again if the flag is not cleared in the handler. On the RX231 device, the interrupt flags (interrupt status flags) are managed internally by the interrupt controller. The interrupt controller has a function that sends an interrupt request to the CPU, DMAC, or DTC and automatically clears the corresponding interrupt status flag when receiving acknowledgment. Therefore, it is unnecessary to perform flag clearing and dummy-read operations that are performed on the H8/3048 device. Note that, in the case of interrupts generated by level detection, the source flags residing in the peripheral modules must be cleared. For details, refer to the “RX230 Group, RX231 Group User's Manual: Hardware”.

![Diagram of Interrupt Processing]

**Figure 1.14** Peripheral Module Interrupt (Generated on Edge Detection) for the H8/3048 Device
Interrupt controller
Peripheral module *2
Interrupt enable bit
CPU/DTC/
DMAC
1. Interrupt request
2. Sends an interrupt request through comparison of priority levels.

Interrupt request enable bit
Interrupt status flag
3. Returns a response indicating that the interrupt was accepted. *1

Compare priority levels

4. Goes to exception handling according to exception vector.

Software interrupt handler
Interrupt processing
RTE

Notes:
1. When a response indicating interrupt request receipt is received from the CPU (or DTC/DMAC), the interrupt flag is cleared automatically. It is not necessary for the handler to clear the flag and do a dummy read.
2. For an interrupt generated by level detection, to clear the interrupt status flag, it is necessary to clear the request flag of the interrupt generation source existing on the peripheral module side.

Figure 1.15 Peripheral Module Interrupt (Generated on Edge Detection) for the RX231 Device
1.8.3 Control of Fast Interrupts
The RX231 device supports fast interrupts in addition to ordinary interrupts.

Ordinary interrupt: After determining the interrupt priority, it is necessary to save the contents of the control registers and general-purpose registers to the internal or external RAM by software.

Fast interrupt: Operation gives the interrupt the highest priority. When a fast interrupt occurs, the contents of the control registers are saved to dedicated registers, thus enabling faster interrupt-triggered activation than an ordinary interrupt.

Some general-purpose registers can be dedicated for interrupts by setting a compiler option. Use of interrupt-dedicated registers makes it unnecessary to perform save and restore operations for general-purpose registers, thus further speeding up fast interrupts.

Note: 1. The save register for fast interrupts is a single-stage register. So, multi-stage fast interrupts are unavailable.
This means that only one function can use fast interrupts.

Figure 1.16 Differences Between Ordinary Interrupts and Fast Interrupts on the RX231 Device
1.8.4 Digital Filter

The RX231 device is provided with a digital filter function for the IRQ and NMI level signals. The sampling clock for the digital filter can be set. Interrupt signals that do not last for at least three cycles of the sampling clock base are not accepted.

![Digital Filter Operation Example](image-url)

**Figure 1.17 Digital Filter Operation Example**
1.8.5 Multiple Interrupts

The RX231 device supports multiple interrupts. To handle multiple interrupts on the RX231 device, it is necessary to set the PSW(I) bit to 1 (enable interrupts) in the interrupt handler. Figure 1.18 shows the sequence of operations performed when multiple interrupts are prohibited. Figure 1.19 shows the sequence of operations performed when multiple interrupts occur.

---

**Figure 1.18** Interrupt Handling Sequence on the RX231 Device (when the PSW(I) bit is cleared)

**Figure 1.19** Interrupt Handling Sequence on the RX231 Device (when the PSW(I) bit is set)
### 2. On-Chip Functions (Peripheral Modules)

#### 2.1 List of On-Chip Functions

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bus controller</td>
<td>Bus controller</td>
</tr>
<tr>
<td>DMA controller</td>
<td>DMA controller (DMACA)</td>
</tr>
<tr>
<td>I/O Ports</td>
<td>I/O Ports</td>
</tr>
<tr>
<td>16-bit integrated timer unit (ITU)</td>
<td>Multifunction timer pulse unit 2 (MTU2a)</td>
</tr>
<tr>
<td>Programmable timing pattern controller (TPC)</td>
<td>—</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>Watchdog timer (WDTA)</td>
</tr>
<tr>
<td>Serial communications interface</td>
<td>Serial communications interface (SCIg/SCIh)</td>
</tr>
<tr>
<td>Smartcard interface</td>
<td></td>
</tr>
<tr>
<td>A/D converter</td>
<td>12-bit A/D converter (S12ADE)</td>
</tr>
<tr>
<td>D/A converter</td>
<td>12-bit D/A converter (R12DAA)</td>
</tr>
<tr>
<td>4 KB RAM</td>
<td>64 KB (max.) RAM</td>
</tr>
<tr>
<td>ROM</td>
<td>ROM</td>
</tr>
<tr>
<td>Mask ROM of 128 KB, max., or flash memory of 128 KB</td>
<td>User area of 512 KB, max.</td>
</tr>
<tr>
<td></td>
<td>Data area of 8 KB</td>
</tr>
<tr>
<td></td>
<td>Extra area for storing the startup area information, access window information, and unique IDs</td>
</tr>
<tr>
<td>Interrupt controller</td>
<td>Interrupt controller (ICUb)</td>
</tr>
<tr>
<td>Low-power mode</td>
<td>Low power consumption</td>
</tr>
<tr>
<td></td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>Function for lower operating power consumption</td>
</tr>
<tr>
<td></td>
<td>Data transfer controller (DTCa)</td>
</tr>
<tr>
<td></td>
<td>Multi-function pin controller (MPC)</td>
</tr>
<tr>
<td></td>
<td>Event link controller (ELC)</td>
</tr>
<tr>
<td></td>
<td>CRC calculator (CRC)</td>
</tr>
<tr>
<td></td>
<td>Serial peripheral interface (RSP1a)</td>
</tr>
<tr>
<td></td>
<td>Serial sound interface (SSI)</td>
</tr>
<tr>
<td></td>
<td>I2C-bus controller (RIICa)</td>
</tr>
<tr>
<td></td>
<td>Port output enable 2 (POE2a)</td>
</tr>
<tr>
<td></td>
<td>Compare match timer (CMT)</td>
</tr>
<tr>
<td></td>
<td>Realtime clock (RTCe)</td>
</tr>
<tr>
<td></td>
<td>Data operation circuit (DOC)</td>
</tr>
<tr>
<td></td>
<td>Clock frequency accuracy measurement circuit (CAC)</td>
</tr>
<tr>
<td></td>
<td>Capacitive touch sensing unit (CTSU)</td>
</tr>
<tr>
<td></td>
<td>SD host interface (SDHIa)</td>
</tr>
<tr>
<td></td>
<td>Memory protection unit (MPU)</td>
</tr>
<tr>
<td></td>
<td>8-bit timer (TMR)</td>
</tr>
<tr>
<td></td>
<td>CAN module (RSCAN)</td>
</tr>
<tr>
<td></td>
<td>Low-power timer (LPT)</td>
</tr>
<tr>
<td></td>
<td>USB 2.0 host/function module (USBd)</td>
</tr>
<tr>
<td></td>
<td>IrDA interface</td>
</tr>
<tr>
<td></td>
<td>Trusted Secure IP (TSIP-Lite)</td>
</tr>
<tr>
<td></td>
<td>Temperature sensor (TEMPSA)</td>
</tr>
<tr>
<td></td>
<td>Comparator B (CMPBa)</td>
</tr>
</tbody>
</table>
2.2 I/O Ports

2.2.1 Comparison of I/O Port Specifications

Table 2.2 Number of I/O Ports

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of pins</td>
<td>100-pin</td>
<td>100-pin</td>
</tr>
<tr>
<td>I/O pins</td>
<td>70</td>
<td>83</td>
</tr>
<tr>
<td>Input pull-up MOS</td>
<td>20</td>
<td>—</td>
</tr>
<tr>
<td>Input pull-up resistor</td>
<td>—</td>
<td>83</td>
</tr>
<tr>
<td>Open-drain output</td>
<td>—</td>
<td>58</td>
</tr>
<tr>
<td>5 V tolerant</td>
<td>—</td>
<td>8</td>
</tr>
<tr>
<td>Schmitt-trigger input</td>
<td>15</td>
<td>—</td>
</tr>
<tr>
<td>Input pins</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>Total number of ports</td>
<td>78</td>
<td>84</td>
</tr>
</tbody>
</table>
2.2.2 I/O Settings

On the H8/3048 and RX231 devices, the ports are shared by multiple functions, such as peripheral module I/O pins and interrupt input pins. Therefore, the functions of the general-purpose input/output ports, peripheral function input/output pins, interrupt input pins, and bus control pins must be selected by configuring the settings of the I/O port registers and on-chip peripheral modules.

On the H8/3048 device, to select the pin for a specific function, a combination of settings for the operating mode and function registers must be specified, and this requires complex operations. In contrast, on the RX231 device, control of pin function selection is centralized in the multi-function pin controller (MPC).

In addition, on the RX231 device, a variety of additional settings can be configured. Figure 2.1 shows an overview of I/O settings on the RX231 device.

![Figure 2.1 Overview of I/O Settings on the RX231 Device](image)

Each I/O port on the RX231 device consists of multiplexed pins. Therefore, the pins of an I/O port must be assigned to the desired port type (a general-purpose I/O port or on-chip module function).

- To use a port as a general-purpose port, assign the port to I/O Port n.*¹
  - Table 2.3 lists the registers that must be configured when an I/O port is to be used as a general-purpose port.
  - Figure 2.2 shows an example of the settings for using a port as a general-purpose port.

- To use a port for a peripheral function, assign the port to I/O Port n and configure the multi-function pin controller (MPC).
  - Table 2.3 to Table 2.6 list the registers that must be configured when an I/O port is to be used for a peripheral function.
  - Figure 2.3 shows the peripheral module initialization sequence.

Note: 1. n indicates a port number that can be 0 to 5, A to E, H, or J.
### Table 2.3 Registers for I/O Ports on the RX231 Device

<table>
<thead>
<tr>
<th>Register</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDR</td>
<td>Port direction register</td>
<td>If a general-purpose I/O port is selected, this register is used to specify whether the port is to be used for input or output.</td>
</tr>
<tr>
<td>PODR</td>
<td>Port output register</td>
<td>A register used for a general-purpose output port. This register stores the output data of the pins to be used for the general-purpose output port.</td>
</tr>
<tr>
<td>PIDR</td>
<td>Port input register</td>
<td>A register used for a general-purpose input port. The pin status of the port is reflected to this register. The pin status can be read from this register regardless of the PDR and PMR values.</td>
</tr>
<tr>
<td>PMR</td>
<td>Port mode register</td>
<td>This register is used to set the pin functions of the port. Whether the pin is used for a general-purpose I/O port or a peripheral function is set for each pin.</td>
</tr>
</tbody>
</table>
| ODR0     | Open drain control register 0 | Selects the port output format from among the following:  
- CMOS output  
- N-channel open drain  
- P-channel open drain  
- Hi-Z |
| ODR1     | Open drain control register 1 | Selects the port output format from among the following:  
- CMOS output  
- N-channel open drain |
| PCR      | Pull-up control register | This register is used to specify whether to enable or disable the input pull-up resistor of the port. |
| PSRA     | Port switching register A | This register is used to select which of the following is to be used:  
- General-purpose I/O function for the PB6 and PB7 pins  
- General-purpose I/O function for the PC0 and PC1 pins |
| PSRB     | Port switching register B | This register is used to select which of the following is to be used:  
- General-purpose I/O function for the PB5, PB3, PB1, and PB0 pins  
- General-purpose I/O function for the PC3, PC2, PC1, and PC0 pins |
| DSCR     | Drive capacity control register | This register is used to specify the drive capacity.  
- Normal drive output  
- High drive output |

**Notes:**  
1. The PSRA register is present on only a 64-pin package product.  
2. The PSRB register is present on only a 48-pin package product.
To assign an I/O port for a peripheral function, configure the registers for the I/O port registers listed in Table 2.3 and the registers listed in Table 2.4 to Table 2.6.

Table 2.4 Register for the Register Write-Protection Function on the RX231 Device

<table>
<thead>
<tr>
<th>Register</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRCR</td>
<td>Protect register</td>
<td>This register is used to control write protection that prevents an accidental rewrite of important registers. This register controls write protection on the following registers: the registers related to the clock generation circuit, the registers related to the operating mode, the registers related to the low power consumption function, the software reset register, the registers related to the low-power timer, the LVD-related registers, and the registers related to the battery back-up function. In addition to the above registers, this register also controls write protection on this register itself.</td>
</tr>
</tbody>
</table>

Table 2.5 Registers Related to the Module Stop Function on the RX231 Device

<table>
<thead>
<tr>
<th>Register</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBYCR</td>
<td>Standby control register</td>
<td>This register is used to control the standby mode.</td>
</tr>
<tr>
<td>MSTPCRA</td>
<td>Module stop control register A</td>
<td>This register is used to disable the following peripheral functions or to control the operating status: TMR, MTU, TPU, CMT, S12AD, DA, DMAC, and DTC</td>
</tr>
<tr>
<td>MSTPCRB</td>
<td>Module stop control register B</td>
<td>This register is used to disable the following peripheral functions or to control the operating status: RSCAN, SCIh, DOC, ELC, comparator B, RSPI0, USB0, RII0, CRC, and SCI0/1/5/6</td>
</tr>
<tr>
<td>MSTPCRC</td>
<td>Module stop control register C</td>
<td>This register is used to disable the following peripheral functions or to control the operating status: RAM, CAC, IRDA, SCI8/9, and deep sleep mode</td>
</tr>
<tr>
<td>MSTPCRD</td>
<td>Module stop control register D</td>
<td>This register is used to disable the following peripheral functions or to control the operating status: CTSU, SSI, SDHI, and Trusted Secure IP</td>
</tr>
<tr>
<td>OPCCR</td>
<td>Operating power consumption control register</td>
<td>This register can be used to set either of the following operating modes: • High-speed operating mode • Middle-speed operating mode</td>
</tr>
<tr>
<td>SOPCCR</td>
<td>Secondary operating power consumption control register</td>
<td>This register can be used to set either of the following operating modes: • High-speed or middle-speed operating mode • Low-speed operating mode</td>
</tr>
<tr>
<td>RSTCKCR</td>
<td>Register for switching the clock source for exiting sleep mode</td>
<td>This register can be used to select the clock source that triggers exiting sleep mode from the following clock sources: • LOCO • HOCO • Main clock oscillator</td>
</tr>
</tbody>
</table>
### Table 2.6 Multi-Function Pin Controller Registers on the RX231 Device

<table>
<thead>
<tr>
<th>Register</th>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PWPR</td>
<td>Write protection register</td>
<td>Function that protects writes to PFS registers</td>
</tr>
<tr>
<td>P0nPFS</td>
<td>P0n pin function control register</td>
<td>Register for selecting the pin function (for port 0)</td>
</tr>
<tr>
<td>P1nPFS</td>
<td>P1n pin function control register</td>
<td>Register for selecting the pin function (for port 1)</td>
</tr>
<tr>
<td>P2nPFS</td>
<td>P2n pin function control register</td>
<td>Register for selecting the pin function (for port 2)</td>
</tr>
<tr>
<td>P3nPFS</td>
<td>P3n pin function control register</td>
<td>Register for selecting the pin function (for port 3)</td>
</tr>
<tr>
<td>P4nPFS</td>
<td>P4n pin function control register</td>
<td>Register for selecting the pin function (for port 4)</td>
</tr>
<tr>
<td>P5nPFS</td>
<td>P5n pin function control register</td>
<td>Register for selecting the pin function (for port 5)</td>
</tr>
<tr>
<td>PAnPFS</td>
<td>PA n pin function control register</td>
<td>Register for selecting the pin function (for port A)</td>
</tr>
<tr>
<td>PBnPFS</td>
<td>PB n pin function control register</td>
<td>Register for selecting the pin function (for port B)</td>
</tr>
<tr>
<td>PCnPFS</td>
<td>PC n pin function control register</td>
<td>Register for selecting the pin function (for port C)</td>
</tr>
<tr>
<td>PDnPFS</td>
<td>PD n pin function control register</td>
<td>Register for selecting the pin function (for port D)</td>
</tr>
<tr>
<td>PEnPFS</td>
<td>PE n pin function control register</td>
<td>Register for selecting the pin function (for port E)</td>
</tr>
<tr>
<td>PHnPFS</td>
<td>PH n pin function control register</td>
<td>Register for selecting the pin function (for port H)</td>
</tr>
<tr>
<td>PAnPFS</td>
<td>PAn pin function control register</td>
<td>Register for selecting the pin function (for port A)</td>
</tr>
<tr>
<td>PBnPFS</td>
<td>PB n pin function control register</td>
<td>Register for selecting the pin function (for port B)</td>
</tr>
<tr>
<td>PCnPFS</td>
<td>PC n pin function control register</td>
<td>Register for selecting the pin function (for port C)</td>
</tr>
<tr>
<td>PDnPFS</td>
<td>PD n pin function control register</td>
<td>Register for selecting the pin function (for port D)</td>
</tr>
<tr>
<td>PEnPFS</td>
<td>PE n pin function control register</td>
<td>Register for selecting the pin function (for port E)</td>
</tr>
<tr>
<td>PHnPFS</td>
<td>PH n pin function control register</td>
<td>Register for selecting the pin function (for port H)</td>
</tr>
</tbody>
</table>

The following figure shows the sequence of initialization operations needed to use an I/O port on the RX231 device as a general-purpose I/O port.

![Sequence of Operations for Using a General-Purpose I/O Port on the RX231 Device](image-url)

**Figure 2.2 Sequence of Operations for Using a General-Purpose I/O Port on the RX231 Device**
Figure 2.3 shows the sequence of initialization operations needed to use an I/O port on the RX231 device for pin functions. For example settings for using peripheral functions, including general-purpose input/output, refer to the relevant chapters.

This flowchart assumes that the pins are initially set for general-purpose input (default).

- Cancels protection. → Disables write protection on the registers related to the low power consumption function.
- Cancels the module stop state for the function module to be used.\(^*1\) (x: A, B, or C)
- Applies protection. → Enables write protection on the registers related to the low power consumption function.
- Sets whether to enable or disable open-drain output/input pull-up resistor, and the drive capacity.
- Sets the initial value to be output from the pin.
- Sets the port direction.
- Sets the pin as a general I/O port pin.
- Enables protection on the PxxPFS register.
- Selects the pin function to be used.
- (If an external bus is to be used) Sets the external bus that corresponds to the CSn#.
- Enables protection on the PxxPFS register.
- Specifies the settings of the registers for the modules to be used.
- Sets the pin as an I/O port for peripheral modules.
- Note that if analog pins are used, PMR remains in general-purpose input mode.

\[\text{Specify necessary module settings}\]

\[\text{Set PMR}\]

\[\text{END}\]

Note: 1. When assigning pin functions to a port, keep in mind that the individual modules are in stopped state by default, except for the DMAC, DTC, and RAM. Therefore, it is necessary to cancel module stop by using the module stop control register (MSTPCRx) of the low power consumption function.

**Figure 2.3 Sequence of Operations for Configuring Pin Functions on the RX231 Device**

Note that the MSTPCRx register is write-protected by the register write protection function when canceling module stop. Therefore, use the protect register (PRCR) to cancel write protection before rewriting the MSTPCRx register. DMAC, DTC, and RAM are operational by default.
2.3 Buses

2.3.1 Comparison of Specifications

Table 2.7 shows the differences in the bus specifications between the H8/3048 and RX231 devices.

Table 2.7 Differences in the Bus Specifications Between the H8/3048 and RX231 Devices

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>External address space</td>
<td>The external address space can be divided into eight areas (areas 0 to 7) and different bus specifications can be set for each area.</td>
<td>The external address space can be divided into four areas (CS0 to CS3) and different bus specifications can be set for each area.</td>
</tr>
<tr>
<td>Area size</td>
<td>128 KB in 1 MB mode<em>1 2 MB in 16 MB mode</em>1</td>
<td>16 MB*2</td>
</tr>
<tr>
<td>Bus width</td>
<td>The 8-bit or 16-bit access space can be selected for each area.</td>
<td>The 8-bit or 16-bit bus space can be selected for each area.</td>
</tr>
<tr>
<td>Bus arbitration function</td>
<td>Each bus master has a fixed priority level:</td>
<td>Whether to use the fixed or toggled priority level can be selected.</td>
</tr>
<tr>
<td></td>
<td>▲ High</td>
<td>The CPU bus, which as a fixed priority level, is an exception.</td>
</tr>
<tr>
<td></td>
<td>External bus master</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Refresh controller</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DMAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CPU</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▼ Low</td>
<td></td>
</tr>
<tr>
<td>Other access control</td>
<td>• Wait modes:</td>
<td>• Wait control</td>
</tr>
<tr>
<td></td>
<td>— Programmable wait mode</td>
<td>— Ability to set the assert/negate timing for chip select signals (CS0# to CS3#)</td>
</tr>
<tr>
<td></td>
<td>— Pin auto-wait mode</td>
<td>— Ability to set assert timing for read signal (RD#) and write signals (WR0#, WR#, and WR1#)</td>
</tr>
<tr>
<td></td>
<td>— Pin wait mode 0</td>
<td>— Ability to set the data output start/stop timing</td>
</tr>
<tr>
<td></td>
<td>— Pin wait mode 1</td>
<td>• Endianness</td>
</tr>
<tr>
<td></td>
<td>Notes:1. The size is common to areas 0 to 7.</td>
<td>— Ability to set endianness separately for each area</td>
</tr>
<tr>
<td></td>
<td>2. The size is common to the CS0 to CS3 areas.</td>
<td></td>
</tr>
</tbody>
</table>
2.3.2 Bus Configuration

This section compares the bus configurations of the H8/3048 and RX231 devices.

The following figure shows the configuration of the bus controller of the H8/3048 device.

![Bus Controller Configuration Diagram]

**Figure 2.4 Configuration of the Bus Controller of the H8/3048 Device**

The H8/3048 device has only internal data buses that are synchronized with the system clock.
The following figure shows the configuration of the buses of the RX231 device.

**Figure 2.5 Configuration of the Buses of the RX231 Device**

The RX231 device has memory buses, internal buses, and peripheral buses, all of which are in a multistage configuration. This configuration allows modules to operate in parallel for each of the CPU, DMAC/DTC, and peripheral buses.

The following lists the buses of the RX231 device. On the RX231 device, as shown in Table 2.8, the clock with which the bus is synchronized differs depending on the bus.

**Table 2.8 Buses of the RX231 Device**

<table>
<thead>
<tr>
<th>Bus</th>
<th>Connected modules</th>
<th>Clock*¹</th>
</tr>
</thead>
</table>
| CPU buses (instruction bus and operand bus) | Instruction bus: CPU (instructions) and on-chip memory  
Operand bus: CPU (operands) and on-chip memory | ICLK |
| Memory bus 1 | On-chip RAM | ICLK |
| Memory bus 2 | On-chip ROM | ICLK |
| Internal main bus 1 | CPU | ICLK |
| Internal main bus 2 | DTC, DMAC, and on-chip memory (RAM and ROM) | ICLK |
| Internal peripheral bus 1 | DTC, DMAC, interrupt controller, and bus error monitoring block | ICLK |
| Internal peripheral bus 2 | Peripheral functions other than internal peripheral buses 1, 3, and 4 | PCLKB |
| Internal peripheral bus 3 | USB0, RSCAN, and CTSU | PCLKB |
| Internal peripheral bus 4 | MTU2 | PCLKA |
| Internal peripheral bus 6 | Flash control module and E2 data flash | FCLK |
| External buses (CS areas) | External devices | BCLK |

Note: 1. ICLK: System clock  
PCLKA: Peripheral module clock A  
PCLKB: Peripheral module clock B  
FCLK: FlashIF clock  
BCLK: External bus clock
2.3.3 Register Configuration
This section shows the registers that are used to configure the buses on the H8/3048 and RX231 devices. On the RX231 device, because it has registers related to bus errors, the CPU can be notified of a bus error to generate an interrupt. Bus errors include invalid address access errors and timeout errors.

Table 2.9 List of Bus Control Registers on the H8/3048 and RX231 Devices

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registers related to CS0 to CS7</td>
<td>Registers related to CS0 to CS3</td>
</tr>
<tr>
<td>Bus width control register (ABWCR)</td>
<td>CSn control register (CSnCR)*1</td>
</tr>
<tr>
<td>Access state control register (ASTCR)</td>
<td>CSn recovery cycle setting register (CSnREC)*1</td>
</tr>
<tr>
<td>Wait control register (WCR)</td>
<td>CS recovery cycle insertion enable register (CSRECE)</td>
</tr>
<tr>
<td>Wait state controller enable register (WCER)</td>
<td>CSn mode register (CSnMOD)*1</td>
</tr>
<tr>
<td>Bus release control register (BRCR)</td>
<td>CSn wait control register 1 (CSnWCR1)*1</td>
</tr>
<tr>
<td>Chip select control register (CSCR)</td>
<td>CSn wait control register 2 (CSnWCR2)*1</td>
</tr>
<tr>
<td>—</td>
<td>Registers related to bus errors</td>
</tr>
<tr>
<td></td>
<td>Bus error status clearing register (BERCLR)</td>
</tr>
<tr>
<td></td>
<td>Bus error monitoring enable register (BEREN)</td>
</tr>
<tr>
<td></td>
<td>Bus error status register 1 (BERSR1)</td>
</tr>
<tr>
<td></td>
<td>Bus error status register 2 (BERSR2)</td>
</tr>
<tr>
<td></td>
<td>Bus priority control register (BUSPRI)</td>
</tr>
</tbody>
</table>

Note: 1. n = 0 to 3

2.3.4 Wait Settings
When configuring bus settings, a wait must be set in accordance with the specifications of the device to be connected. To set a wait on the H8/3048 device, select an appropriate wait mode. Table 2.10 lists the wait modes available on the H8/3048 device. To set a wait on the RX231 device, configure the registers listed in Table 2.11.

Table 2.10 Wait Modes Available on the H8/3048 Device

<table>
<thead>
<tr>
<th>Wait Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin wait mode 0</td>
<td>Wait states can be set from only the #WAIT pin.</td>
</tr>
<tr>
<td>Programmable wait</td>
<td>Wait states are always inserted as configured with the WC1 and WC0 bits in the WCR register. Wait states cannot be inserted from the #WAIT pin.</td>
</tr>
<tr>
<td>Pin wait mode 1</td>
<td>Wait states are always inserted as configured with the WC1 and WC0 bits in the WCR register. In addition, wait states can also be inserted from the #WAIT pin.</td>
</tr>
<tr>
<td>Pin auto-wait mode</td>
<td>Wait states are inserted from the #WAIT pin as configured with the WC1 and WC0 bits in the WCR register.</td>
</tr>
</tbody>
</table>
### Table 2.11 Setting Items for the External Bus Interface Registers on the RX231 Device (CS Area Settings)

<table>
<thead>
<tr>
<th>Register</th>
<th>Symbol</th>
<th>Setting Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSnMOD*1</td>
<td>EWENB</td>
<td>Whether to enable an external wait*2</td>
</tr>
<tr>
<td>CSnWCR1**1</td>
<td>CSPWWAIT</td>
<td>Page write cycle wait</td>
</tr>
<tr>
<td></td>
<td>CSPRWAIT</td>
<td>Page read cycle wait</td>
</tr>
<tr>
<td></td>
<td>CSWWAIT</td>
<td>Normal write cycle wait</td>
</tr>
<tr>
<td></td>
<td>CSRWAIT</td>
<td>Normal read cycle wait</td>
</tr>
<tr>
<td>CSnWCR2**1</td>
<td>CSROFF</td>
<td>Read-access CS extension cycle wait</td>
</tr>
<tr>
<td></td>
<td>CSWOFF</td>
<td>Write-access CS extension cycle wait</td>
</tr>
<tr>
<td></td>
<td>WDOFF</td>
<td>Write data output CS extension cycle wait</td>
</tr>
<tr>
<td></td>
<td>AWAIT</td>
<td>Address cycle wait</td>
</tr>
<tr>
<td></td>
<td>RDON</td>
<td>RD assert wait</td>
</tr>
<tr>
<td></td>
<td>WRON</td>
<td>WR assert wait</td>
</tr>
<tr>
<td></td>
<td>WDON</td>
<td>Write data output wait</td>
</tr>
<tr>
<td></td>
<td>CSON</td>
<td>CS assert wait</td>
</tr>
<tr>
<td>CSnREC*1</td>
<td>RRCV</td>
<td>Read recovery cycle</td>
</tr>
<tr>
<td></td>
<td>WRCV</td>
<td>Write recovery cycle</td>
</tr>
<tr>
<td>CSRECEM</td>
<td>RCVEN0 to RCVEN7</td>
<td>Separate bus recovery cycle insertion enable 0 to 7</td>
</tr>
<tr>
<td></td>
<td>RCVENM0 to RCVENM7</td>
<td>Multiplexed bus recovery cycle insertion enable 0 to 7</td>
</tr>
</tbody>
</table>

**Notes:**
1. \( n = 0 \) to 3
2. Wait cycles can be inserted using the WAIT# signal by enabling external wait.

Make sure that the set value of the wait cycle does not violate the limitations described in the hardware manual for the RX231 Group. For the limitations to follow when setting the value, refer to:

- **RX230 Group, RX231 Group User’s Manual: Hardware (R01UH0496EJ)**  
  Section 16.5.7, “Limitations”

#### 2.3.5 External Bus Pin Setting Example

For details on how to configure the settings for the external bus pins, refer to:

- **RX230 Group, RX231 Group User’s Manual: Hardware (R01UH0496EJ)**  
  Section 22.3, “Settings for the External Bus Interface”
2.4 Interrupt Controller

2.4.1 Usage Example for IRQ Pins

The following shows an example of settings for using the IRQ3 pin. On the H8/3048 device, P83 is used for the IRQ3 input pin. On the RX231 device, P33 is used for the IRQ3 input pin.

<table>
<thead>
<tr>
<th>Table 2.12 Interrupt Initial Setting Example (IRQ3 Settings)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Procedure</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
## 2.5 DMA Controller

### 2.5.1 Comparison of Specifications

This section compares the DMA controller specifications of the H8/3048 and RX231 devices.

#### Table 2.13 Differences in the DMAC Specifications Between the H8/3048 and RX231 Devices (1/2)

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of channels</strong></td>
<td>4 channels (short address mode) or 2 channels (full address mode)</td>
<td>4 channels (DMACm (m = 0 to 3))</td>
</tr>
<tr>
<td><strong>Transfer space</strong></td>
<td>The 16-MB address space can be directly accessed.</td>
<td>512 MB (areas within the range from 00000000h to 0FFFFFFFh and the range from F0000000h to FFFFFFFFh, excluding reserved areas)</td>
</tr>
<tr>
<td><strong>Maximum number of data units that can be transferred</strong></td>
<td>16 M data units (= Maximum number of data units that can be transferred in block transfer mode (255) × Maximum transfer count (65,536))</td>
<td>1 M data units (= Maximum number of data units that can be transferred in block transfer mode (1,024) × Number of blocks (1,024))</td>
</tr>
</tbody>
</table>
| **DMA activation sources**    | • Internal interrupt  
• External request  
• Auto request | The activation source can be selected for each channel.  
• Software trigger  
• Interrupt request from a peripheral module  
• Trigger input to an external interrupt input pin |
| **Channel priority**          | • Short address:  
  Channel 0A > channel 0B > channel 1A > channel 1B  
  (Channel 0A has the highest priority.)  
  • Full address:  
  Channel 0 > channel 1  
  (Channel 0 has the highest priority.) | • Channel 0 > channel 1 > channel 2 > channel 3  
  (Channel 0 has the highest priority.) |
| **Transfer data**             | 1 data unit    | Bit length: 8 or 16 bits | Bit length: 8, 16, or 32 bits |
| **Block size**                | 1 to 255 bytes | Number of data units: 1 to 1,024 |
| **Transfer mode**             | Normal transfer mode  
• Repeat transfer mode*1  
• Block transfer mode  
• I/O mode*1  
• Idle mode*1 | Normal transfer mode  
• Repeat transfer mode  
• Block transfer mode |
### Differences in the DMAC Specifications Between the H8/3048 and RX231 Devices (2/2)

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048 DMAC</th>
<th>RX231 DMACA</th>
</tr>
</thead>
</table>
| DMA interrupt request | • Issued when as many data units as specified by the transfer counter are transferred. | • Transfer end interrupt  
  — Generated when as many data units as specified by the transfer counter are transferred.  
  • Transfer escape end interrupt  
  — Generated when as much data as the repeat size is transferred.  
  — Generated when the extended repeat area overflows. |
| Low power consumption | The modules can be placed in standby status.                                  | The modules can be placed in stopped state.                                  |
| Other items           | —                                                                           | • Extended repeat area function  
  • Event link function                                                      |

**Note:** 1. This is a transfer mode available in short address mode.
2.5.2 DMAC Block Diagram
The following figure shows the block diagram for the DMAC of the H8/3048 device.

![DMAC Block Diagram](image)

Legend:
- DTCR: Data transfer control register
- MAR: Memory address register
- IOAR: I/O address register
- ETCR: Transfer count register

Figure 2.6 Block Diagram for the DMAC of the H8/3048 Device
The following figure shows the block diagram for the DMACA of the RX231 device.

![Block Diagram for the DMACA of the RX231 Device](image)

**Figure 2.7** Block Diagram for the DMACA of the RX231 Device
### 2.5.3 List of Registers

The following table lists the registers for the DMA controllers of the H8/3048 and RX231 devices.

**Table 2.14 List of Registers for the DMA Controller**

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DMAC</strong></td>
<td><strong>DMACA</strong></td>
</tr>
<tr>
<td>Memory address register</td>
<td>DMA module start register (DMAST)</td>
</tr>
<tr>
<td>Short address mode: MARkm</td>
<td>DMA transfer source register n (DMACn.DMSAR)</td>
</tr>
<tr>
<td>Full address mode: MARkm and MARkm</td>
<td>DMA transfer destination register n (DMACn.DMDAR)</td>
</tr>
<tr>
<td><strong>I/O address register</strong></td>
<td></td>
</tr>
<tr>
<td>Short address mode: IOARkm</td>
<td>DMA transfer count register n (DMACn.DMCRA)</td>
</tr>
<tr>
<td>Full address mode: None</td>
<td>DMA block transfer count register n (DMACn.DMCRB)</td>
</tr>
<tr>
<td><strong>Transfer count register</strong></td>
<td></td>
</tr>
<tr>
<td>Short address mode: ETCRkm</td>
<td>DMA address mode register n (DMACn.DMAMD)</td>
</tr>
<tr>
<td>Full address mode: ETCRkm and ETCRkm</td>
<td>DMA offset register (DMACn.DMOMFR)</td>
</tr>
<tr>
<td><strong>Data transfer control register</strong></td>
<td></td>
</tr>
<tr>
<td>Short address mode: DTCRkm</td>
<td>DMA transfer enable register n (DMACn.DMCNT)</td>
</tr>
<tr>
<td>Full address mode: DTCRkm and DTCRkm</td>
<td>DMA software start register n (DMACn.DMREQ)</td>
</tr>
<tr>
<td>—</td>
<td>DMA status register n (DMACn.DMST)</td>
</tr>
<tr>
<td></td>
<td>DMA activation source flag control register n (DMACn.DMCSL)</td>
</tr>
</tbody>
</table>

*Note: k = 0 or 1, m = A or B, n = 0 to 3*
### 2.5.4 DMAC Activation Sources

Table 2.15 compares the DMA controller activation sources of the H8/3048 and RX231 devices.

<table>
<thead>
<tr>
<th>Activation Sources</th>
<th>H8/3048 DMAC</th>
<th>RX231 DMACA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Short address mode</td>
<td>Full address mode</td>
</tr>
<tr>
<td></td>
<td>Channel 0A, 1A</td>
<td>Channel 0B, 1B</td>
</tr>
</tbody>
</table>

|                      | Internal interrupt | High-function timer | ITU compare match/ input capture interrupt (H8/3048) | IMIA0 to IMIA3 | ○ | × | ○ |
|                      | MTU compare match/ input capture interrupt (RX231) | TGIA0 to TGIA4 | ○ |
|                      | TPU compare match/ input capture (RX231) | TGIA0A to TGIA5A | ○ |
|                      | SCI0 Transmit data empty interrupt | TXI0 | ○ | × | ○ |
|                      | SCI0 Receive data full interrupt | RXI0 | ○ | × | ○ |
|                      | Interrupts from peripheral modules other than the above | On the RX231 device, interrupts other than the above can also be set as DMACA activation sources. These interrupts include IRQ, TXI1 to TXI12, and RXI1 to RXI12. For details, refer to the interrupt vector table in section 15.3.1 in the hardware manual for the RX231 Group | ○ | × | ○ |
|                      | External request | Falling edge of the DREQ pin | ○ | ○ | ○ | ○ | ○ |
|                      | Auto-request | Low-level input on the DREQ pin | ○ | ○ | ○ | × |

○: Available as an activation source  
×: Unavailable as an activation source  

**Note:** 1. Request signals from an external device cannot be used to initiate transfers directly using the DMACA of the RX231 device. To use a request signal from an external device to active the DMAC, set IRQ as the DMAC activation source and input the request signal to the IRQ pin. 

- **Activation source setting method:**
  - On the H8/3048 device, set the DMAC activation source in the DTCR register.
  - On the RX231 device, set the DMACA activation source in DMRSRm*1 for the interrupt controller.

**Note:** 1. m: DMACA channel number (0 to 3)
2.5.5 Transfer Modes
Table 2.16 shows the transfer mode specifications of the DMA controllers of the H8/3048 and RX231 devices.

Table 2.16 Transfer Mode Specifications of the DMA Controller

<table>
<thead>
<tr>
<th>Transfer Mode</th>
<th>Transfer Size</th>
<th>Transfer Count</th>
<th>How the Transfer Source and Destination Addresses Are Updated after a Requested Transfer Is Complete</th>
</tr>
</thead>
</table>
| H8/3048 DMAC  | Short address mode | 1 byte | 65,536 • IOAR registers: Non-change  
• MAR registers: Increment, decrement, or non-change can be selected. |
|               | Idle mode     | 1 word | 65,536 • IOAR and MAR registers: Non-change |
|               | Repeat mode   |       | 256 • IOAR registers: Non-change  
• MAR registers: Increment, decrement, or non-change can be selected. |
|               | Normal mode   |       | 65,536 • MARA and MARB registers: Increment, decrement, or non-change can be selected. |
|               | Block transfer mode (Maximum number of data units per block: 255) |       | 65,536 |
| RX231 DMACA   | Normal transfer mode | 1 byte | 65,535 DMSAR and DMDAR registers: Increment, decrement, non-change, or increment-by-offset (DMAC0 only) can be selected for each register. |
|               | Repeat transfer mode (Maximum repeat size: 1024 data units) | 1 word | 1,024 |
|               | Block transfer mode (Maximum number of data units per block: 1024) | 1 longword | |

Note: Supplementary information regarding the repeat mode on the H8/3048 device and the repeat transfer mode on the RX231 device:

In repeat mode on the H8/3048 device, the data in the range set by the transfer count is transferred repeatedly until the CPU clears the DTE bit to 0.

In repeat transfer mode on the RX231 device, a DMA transfer ends when data of the repeat size is transferred the number of times specified by the repeat count. The DMA transfer can be resumed by writing “1” to the DMACm.DMCNT.DTE bit in the handler of the repeat size end interrupt.
Figure 2.8 shows an overview of the DMACA operation on the RX231 device in each transfer mode.

- Normal transfer mode
  - In normal transfer mode, a single unit of data is transferred per transfer request. This transfer can be performed a maximum of 65,535 times. The address update (increment, decrement, non-change, or increment-by-offset) performed after a transfer can be controlled using DMSAR and DMDAR registers separately. In the operation example shown in Figure 2.8, the values of the DMSAR and DMDAR registers are incremented.

- Repeat transfer mode

- Block transfer mode
  - Block area
    - Block 1
    - Block 2
    - Block n
• Repeat transfer mode
  In repeat transfer mode, a single unit of data is transferred per transfer request. In this mode, a maximum of 1,024 can be set for the repeat size and repeat count. The address set in the address register (DMSAR or DMDAR) assigned for the repeat area is reset to the initial address when data of the repeat size is transferred. When this processing is repeated the number of times specified by the repeat count, a DMA transfer ends. The DMA transfer can be resumed by writing “1” to the DMACm.DMCNT.DTE bit in the handler of the repeat size end interrupt. In the operation example shown in Figure 2.8, destination address register is assigned for the repeat area.

• Block transfer mode
  In block transfer mode, a single block (a maximum of 1 KB) of data is transferred per transfer request. This transfer can be performed a maximum of 1,024 times. The block size is set in the DMCRA register. The DTS bit of the MRB register is used to set whether the source address register or destination address register is assigned for the block area. In the operation example in Figure 2.8, the source address register is assigned for the block area.

2.5.6 Module Stop
On the RX231 device, after recovery from a reset, modules other than the DMAC, DTC, and RAM are placed in stopped state (clock supply is stopped) to reduce power consumption. The DMAC can also be stopped for further reduction of power consumption by using the module stop control register. In this case, however, the DTC module is also stopped.
2.6 Serial Communications Interface

2.6.1 Comparison of Specifications

Whereas the H8/3048 device provides only one serial communications interface (SCI), the RX231 device provides two: SCIg and SCIh. SCIg supports the smartcard (IC card) interface as an extended function of the asynchronous mode, in addition to conventional transfer modes, asynchronous mode and clock-synchronous mode. SCIg also supports the simplified I2C bus interface (single master operation) and simplified SPI bus interface. SCIh has all functions of SCIg and extended serial interface functions. For details on the transfer modes not available on the H8/3048 device, refer to the “RX230 Group, RX231 Group User’s Manual: Hardware”.

Table 2.17 Differences in the Serial Communications Interface Between the Devices (1/2)

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of channels</td>
<td>2 (SCI0 and SCI1)</td>
<td>7 (SCIg: SCI0, SCI1, SCI5, SCI6, SCI8, and SCI9; SCIh: SCI12)</td>
</tr>
<tr>
<td>Serial communication modes</td>
<td>Asynchronous</td>
<td>Asynchronous</td>
</tr>
<tr>
<td></td>
<td>Clock-synchronous</td>
<td>Clock-synchronous</td>
</tr>
<tr>
<td></td>
<td>Smartcard interface (SCI0 only)</td>
<td>Smartcard interface</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Simplified I2C bus</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Simplified SPI bus</td>
</tr>
<tr>
<td>Transfer rate</td>
<td>The desired bit rate can be</td>
<td></td>
</tr>
<tr>
<td></td>
<td>selected using the on-chip baud</td>
<td></td>
</tr>
<tr>
<td></td>
<td>rate generator.</td>
<td></td>
</tr>
<tr>
<td>Full duplex communication</td>
<td>Transmit block: A dual-buffer</td>
<td>Receive block: A dual-buffer</td>
</tr>
<tr>
<td></td>
<td>configuration allows continuous</td>
<td>configuration allows continuous</td>
</tr>
<tr>
<td></td>
<td>transmission.</td>
<td>reception.</td>
</tr>
<tr>
<td>Data transfer</td>
<td>Only LSB-first is supported.</td>
<td>LSB-first or MSB-first can be</td>
</tr>
<tr>
<td></td>
<td></td>
<td>selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(The simplified I2C bus supports</td>
</tr>
<tr>
<td></td>
<td></td>
<td>only MSB-first.)</td>
</tr>
<tr>
<td>Interrupt source</td>
<td>Transmit data empty</td>
<td>Transmit data empty</td>
</tr>
<tr>
<td></td>
<td>Transmit end</td>
<td>Transmit end</td>
</tr>
<tr>
<td></td>
<td>Receive data full</td>
<td>Receive data full</td>
</tr>
<tr>
<td></td>
<td>Receive error</td>
<td>Receive error</td>
</tr>
<tr>
<td></td>
<td>End of generating start, restart,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>or stop conditions (for simplified</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I2C mode only)</td>
<td></td>
</tr>
<tr>
<td>Low power consumption</td>
<td>The module standby state can be</td>
<td>The module stop state can be</td>
</tr>
<tr>
<td></td>
<td>set for each channel.</td>
<td>set for each channel.</td>
</tr>
<tr>
<td>Asynchronous mode</td>
<td>Data length</td>
<td>7, 8, or 9 bits</td>
</tr>
<tr>
<td></td>
<td>7 or 8 bits</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 or 2 bits</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Parity</td>
<td>Even, odd, or none</td>
</tr>
<tr>
<td></td>
<td>Receive error detection</td>
<td>Parity, overrun, and framing</td>
</tr>
<tr>
<td></td>
<td>Hardware flow control</td>
<td>Not implemented</td>
</tr>
<tr>
<td></td>
<td>Start bit detection</td>
<td>Implemented (controllable using</td>
</tr>
<tr>
<td></td>
<td>Break detection</td>
<td>the CTSn and RTSn pins)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Low level or falling edge can be</td>
</tr>
<tr>
<td></td>
<td></td>
<td>selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A break can be detected by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>directly reading the level of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RxDn pin when a framing error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>occurs.</td>
</tr>
</tbody>
</table>
## Table 2.17 Differences in the Serial Communications Interface Between the Devices (2/2)

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asynchronous mode</td>
<td>Clock source</td>
<td>The internal or external clock can be selected. The transfer rate clock can be input from TMR (channel SCI5 or SCI6 only).</td>
</tr>
<tr>
<td></td>
<td>Double-speed mode</td>
<td>Not implemented</td>
</tr>
<tr>
<td></td>
<td>Multiprocessor communication</td>
<td>Implemented</td>
</tr>
<tr>
<td></td>
<td>Digital noise filter</td>
<td>Not implemented</td>
</tr>
<tr>
<td>Clock synchronous mode</td>
<td>Data length</td>
<td>8 bits</td>
</tr>
<tr>
<td></td>
<td>Receive error detection</td>
<td>Overrun errors</td>
</tr>
<tr>
<td></td>
<td>Hardware flow control</td>
<td>Not implemented</td>
</tr>
<tr>
<td>Smartcard interface</td>
<td>Implemented</td>
<td></td>
</tr>
<tr>
<td>Simplified I2C mode</td>
<td>Not implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>Simplified SPI mode</td>
<td>Not implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>Extended serial mode</td>
<td>Not implemented</td>
<td>Implemented on SCIh (SCI12) only</td>
</tr>
<tr>
<td>Event link function</td>
<td>Not implemented</td>
<td>Available with SCI5 only.</td>
</tr>
</tbody>
</table>

The following table compares the on-chip SCI registers of the devices and shows the differences in these registers.

## Table 2.18 List of SCI Registers (1/2)

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmit data register (TDR)</td>
<td>Transmit data register (TDR)</td>
<td>None</td>
</tr>
<tr>
<td>Transmit shift register (TSR)</td>
<td>Transmit shift register (TSR)</td>
<td></td>
</tr>
<tr>
<td>Receive data register (RDR)</td>
<td>Receive data register (RDR)</td>
<td></td>
</tr>
<tr>
<td>Receive shift register (RSR)</td>
<td>Receive shift register (RSR)</td>
<td></td>
</tr>
<tr>
<td>Serial mode register (SMR)</td>
<td>Serial mode register (SMR)</td>
<td></td>
</tr>
<tr>
<td>Serial control register (SCR)</td>
<td>Serial control register (SCR)</td>
<td></td>
</tr>
<tr>
<td>Serial status register (SSR)</td>
<td>Serial status register (SSR)</td>
<td></td>
</tr>
<tr>
<td>Bit rate register (BRR)</td>
<td>Bit rate register (BRR)</td>
<td></td>
</tr>
<tr>
<td>Smart card mode register (SCMR)</td>
<td>Smart card mode register (SCMR)</td>
<td></td>
</tr>
</tbody>
</table>
### Table 2.18 List of SCI Registers (2/2)

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>— (There is no corresponding register.)</td>
<td>Modulation duty register (MDDR)</td>
<td>New register added on the RX231</td>
</tr>
<tr>
<td></td>
<td>Serial extended mode register (SEMR)*1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Noise filter setting register (SNFR)*1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I²C mode registers 1 to 3 (SIMR1 to SIMR3)*1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>I²C status register (SISR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SPI mode register (SPMR)*1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Extended serial mode enable register (ESMER)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Control registers 0 to 3 (CR0 to CR3)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port control register (PCR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Interrupt control register (ICR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Status register (STR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Status clearing register (STCR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Control field 0 data register (CF0DR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Control field 0 compare enable register (CF0CR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Control field 0 receive data register (CF0RR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Primary control field 1 data register (PCF1DR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Secondary control field 1 data register (SCF1DR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Control field 1 compare enable register (CF1CR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Control field 1 receive data register (CF1RR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Timer control register (TCR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Timer mode register (TMR)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Timer prescaler register (TPRE)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Timer counter register (TCNT)</td>
<td></td>
</tr>
</tbody>
</table>

Note: 1. Some bit fields in these registers are used for asynchronous or clock-synchronous serial communication settings.
2.6.2 SCI-related Settings Required During Migration

For the processing that performed serial communication in asynchronous mode or clock-synchronous mode by using the SCI on the H8/3048 device, the following settings must be configured on the RX231 device:

- Determining the one-bit period and selecting the clock source
  To perform communication in asynchronous mode, use the serial extended mode register (SEMR) to select the external clock input or TM clock input (channel SCI5, SCI6, or SCI12 only) as the clock source for determining the one-bit period. Also select 8 or 16 as the number of base clock cycles per one-bit period.

- Digital noise filter
  Whether to enable or disable the digital noise filter is set by using the serial extended mode register (SEMR). If the noise filter is enabled, use the noise filter setting register (SNFR) to select the clock for the noise filter.

2.6.3 Module Stop

On the RX231 device, after recovery from a reset, modules other than the DMAC, DTC, and RAM are placed in stopped state (clock supply is stopped) to reduce power consumption. Before configuring the modules, make sure that they are not in stopped state.
2.7 High-Function Timers (ITU, MTU2a, and TPUa)

2.7.1 Comparison of Specifications

The MTU2a and TPUa on the RX231 device are based on the ITU of the H8/3048 device, and provide more advanced features. Some setting items supported by the ITU are not available on the TPUa but they are available with the MTU2a. For I/O pins, as shown in Table 2.20, the MTU2a and TPUa on the RX231 device have all I/O pins of the ITU on the H8/3048 device.

Table 2.19 Comparison of Specifications (Functions) of the ITU on the H8/3048 Device, and the MTU2a and TPUa on the RX231 Device (1/2)

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048 ITU</th>
<th>RX231 MTU2a</th>
<th>TPUa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input clock Internal clock</td>
<td>Internal clock: φ, φ/2, φ/4, and φ/8</td>
<td>MTU2a and TPUa support: PCLK/1, PCLK/4, PCLK/16, PCLK/64, PCLK/256, and PCLK/1024</td>
<td>TPUa also supports: PCLK/4096</td>
</tr>
<tr>
<td>External clock</td>
<td>TCLKA, TCLKB, TCLKC, and TCLKD</td>
<td>MTCLKA, MTCLKB, MTCLKC, and MTCLKD</td>
<td>TCLKA, TCLKB, TCLKC, and TCLKD</td>
</tr>
<tr>
<td>Number of channels</td>
<td>5 (channels 0 to 4)</td>
<td>6 (MTU0 to MTU5)</td>
<td>6 (TPU0 to TPU5)</td>
</tr>
<tr>
<td>Buffer register</td>
<td>Supported on channels 3 and 4 only</td>
<td>Supported on MTU0, MTU3, and MTU4 only</td>
<td>Supported on TPU0 and TPU3 only</td>
</tr>
<tr>
<td>Counter clearing source</td>
<td>Compare match and input capture</td>
<td>Compare match and input capture</td>
<td>Compare match and input capture</td>
</tr>
<tr>
<td>Compare match output</td>
<td>0, 1, or Toggle (Toggle is not supported on channel 2.)</td>
<td>0, 1, or Toggle</td>
<td>0, 1, or Toggle</td>
</tr>
<tr>
<td>Synchronization</td>
<td>Channels 0 to 4</td>
<td>MTU0 to MTU4</td>
<td>TPU0 to TPU5</td>
</tr>
<tr>
<td>PWM mode 1</td>
<td>Channels 0 to 4</td>
<td>MTU0 to MTU4</td>
<td>TPU0 to TPU5</td>
</tr>
<tr>
<td>Complementary PWM mode</td>
<td>Channels 3 and 4</td>
<td>MTU3 and MTU4</td>
<td>—</td>
</tr>
<tr>
<td>Reset-synchronized PWM mode</td>
<td>Channels 3 and 4</td>
<td>MTU3 and MTU4</td>
<td>—</td>
</tr>
<tr>
<td>Phase counting mode</td>
<td>Channel 2</td>
<td>MTU1 and MTU2</td>
<td>TPU1, TPU2, TPU4, and TPU5</td>
</tr>
<tr>
<td>DMAC activation</td>
<td>Channels 0 to 3</td>
<td>MTU0 to MTU4</td>
<td>TPU0 to TPU5</td>
</tr>
<tr>
<td>Number of interrupt sources</td>
<td>15</td>
<td>28</td>
<td>26</td>
</tr>
<tr>
<td>Low power consumption</td>
<td>The modules can be placed in standby state.</td>
<td>The modules can be placed in stopped state.</td>
<td>The modules can be placed in stopped state.</td>
</tr>
<tr>
<td>PWM mode 2</td>
<td>—</td>
<td>MTU0 to MTU2</td>
<td>—</td>
</tr>
<tr>
<td>AC synchronous motor drive mode</td>
<td>—</td>
<td>MTU0, MTU3, and MTU4</td>
<td>—</td>
</tr>
<tr>
<td>DTC activation</td>
<td>—</td>
<td>MTU0 to MTU5</td>
<td>TPU0 to TPU4</td>
</tr>
<tr>
<td>A/D conversion start trigger</td>
<td>—</td>
<td>MTU0 to MTU4</td>
<td>TPU0 to TPU4</td>
</tr>
<tr>
<td>Event link function (output)</td>
<td>—</td>
<td>20 sources (MTU1 to MTU4)</td>
<td>—</td>
</tr>
</tbody>
</table>
Table 2.19 Comparison of Specifications (Functions) of the ITU on the H8/3048 Device, and the MTU2a and TPUa on the RX231 Device (2/2)

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ITU</td>
<td>MTU2a</td>
</tr>
<tr>
<td>Event link function (input)</td>
<td>—</td>
<td>MTU1 to MTU4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Count-start operation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Input-capture operation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Count-restart operation</td>
</tr>
<tr>
<td>A/D converter start request delaying function</td>
<td>—</td>
<td>MTU4</td>
</tr>
<tr>
<td>Interrupt skipping</td>
<td>—</td>
<td>MTU3 and MTU4</td>
</tr>
</tbody>
</table>

Table 2.20 Comparison of Specifications (Pins) of the ITU on the H8/3048 Device, and the MTU2a and TPUa on the RX231 Device

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ITU</td>
<td>MTU2a</td>
</tr>
<tr>
<td>I/O pins</td>
<td>10 pins</td>
<td>16 pins (max.)</td>
</tr>
<tr>
<td>Output pins</td>
<td>2 pins (channel 4)</td>
<td>—</td>
</tr>
<tr>
<td>Input pins</td>
<td>—</td>
<td>3 pins (MTU5)</td>
</tr>
</tbody>
</table>

2.7.2 Notes on Replacing the Timer

- If the ITU is to be replaced by the MTU2a, note that the interrupt flags for the timer status registers (TSR) are located in the interrupt controller.
- If the ITU is to be replaced by the TPUa, the interrupt flags for the timer status registers (TSR) are also located in the interrupt controller, although the timer status registers (TSR) themselves have their own interrupt flags.
## 2.7.3 List of Registers

### Table 2.21 List of Registers for the ITU, MTU2a, and TPUa (1/2)

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSTR</td>
<td>MTU.TSTR</td>
<td>TPU.TSTR</td>
</tr>
<tr>
<td>TSNC</td>
<td>MTU.TSYR</td>
<td>TPU.TSYR</td>
</tr>
<tr>
<td>TMDR</td>
<td>MTU0.TMDR to MTU4.TMDR</td>
<td>TPU0.TMDR to TPU5.TMDR</td>
</tr>
<tr>
<td>TFCR</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>TOER</td>
<td>MTU.TOER</td>
<td>—</td>
</tr>
<tr>
<td>TCNT0 to TCNT4</td>
<td>MTU0.TCNT to MTU4.TCNT, MTU5.TCNTU, MTU5.TCNTV, and MTU5.TCNTW</td>
<td>TPU0.TCNT to TPU5.TCNT</td>
</tr>
<tr>
<td>GRA0 to GRA4 and GRB0 to GRB4</td>
<td>MTU0.TGRA to MTU0.TGRF, MTU1.TGRA, MTU1.TGRB, MTU2.TGRA, MTU2.TGRB, MTU3.TGRA to MTU3.TGRD, MTU4.TGRA to MTU4.TGRD, MTU5.TGRU, MTU5.TGRV, and MTU5.TGRW</td>
<td>TPU0.TGRA to TPU0.TGRD, TPU1.TGRA, TPU1.TGRB, TPU2.TGRA, TPU2.TGRB, TPU3.TGRA to TPU3.TGRD, TPU4.TGRA, TPU4.TGRB, TPU5.TGRA, and TPU5.TGRB</td>
</tr>
<tr>
<td>BRA3, BRA4, BRB3, and BRB4</td>
<td>MTU0.TGRC, MTU3.TGRC, MTU4.TGRC, MTU0.TGRD, MTU3.TGRD, MTU4.TGRD, and MTU0.TGFR*2, *4</td>
<td>TPU0.TGRC, TPU3.TGRC, TPU0.TGRD, and TPU3.TGRD*3, *4</td>
</tr>
<tr>
<td>TOCR</td>
<td>MTU.TOCR1</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>MTU.TOCR2</td>
<td>—</td>
</tr>
<tr>
<td>TCR0 to TCR4</td>
<td>MTU0.TCR to MTU4.TCR, MTU5.TCRU, MTU5.TCRV, and TCRW</td>
<td>TPU0.TCR to TPU5.TCR</td>
</tr>
<tr>
<td>TIOR0 to TIOR4</td>
<td>MTU0.TIORH, MTU0.TIORL, MTU1.TIOR, MTU2.TIOR, MTU3.TIORH, MTU3.TIORL, MTU4.TIORH, and MTU4.TIORL</td>
<td>TPU0.TIORH, TPU0.TIORL, TPU1.TIOR, TPU2.TIOR, TPU3.TIORH, TPU3.TIORL, TPU4.TIOR, and TPU5.TIOR</td>
</tr>
<tr>
<td>TSR0 to TSR4</td>
<td>—</td>
<td>TPU0.TSR to TPU5.TSR</td>
</tr>
<tr>
<td>TIER0 to TIER4</td>
<td>MTU0.TIER to MTU4.TIER</td>
<td>TPU0.TIER to TPU5.TIER</td>
</tr>
</tbody>
</table>
### Table 2.21 List of Registers for the ITU, MTU2a, and TPUa (2/2)

<table>
<thead>
<tr>
<th>ITU</th>
<th>RX231</th>
<th>TPUa</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>MTU2a.TSR to MTU4.TSR</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU0.NFCR to MTU4.NFCR</td>
<td>TPU0.NFCR to TPU5.NFCR</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU5.TCNTCMPCLR</td>
<td>TPU0.NFCR to TPU5.NFCR</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU1.TICCR</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TRWER</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TGCR</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU0.TBTM, MTU3.TBTM, and MTU4.TBTM</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TOLBR</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TCNTS</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TDDR</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TCBR</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TDER</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TWCR</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU4.TADCR</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU4.TADCORA and MTU4.TADCORB</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU4.TADCOBRA and MTU4.TADCOBRB</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TITCR</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TITCNT</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>MTU.TBTER</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

Notes:
1. Bit assignments are different.
2. The TGRC and TGRD registers for the MTU0, MTU3, and MTU4 modules, and the TRGF register for the MTU0 module can be set to operate as buffer registers.
3. The TGRA and TGRB registers for the TPU0 and TPU3 modules can be set to operate as buffer registers.
4. Possible combinations of TGR registers and buffer registers are as follows: TGRA-TGRC, TGRB-TGRD, and TGRE-TGRF

#### 2.7.4 Module Stop

On the RX231 device, after recovery from a reset, modules other than the DMAC, DTC, and RAM are placed in stopped state (clock supply is stopped) to reduce power consumption. Before configuring the modules, make sure that they are not in stopped state.
2.8 Timer-based Pulse Output Function

2.8.1 Comparison of Specifications

The RX231 device does not have a module equivalent to the programmable timing pattern controller (TPC) on the H8/3048 device. This section shows a method for achieving the same functionality as the TPC on the RX231 device. Note that the RX651 or RX65N device has a programmable pulse generator (PPG), which can substitute for the TPC on the H8/3048 device. For details, refer to the “RX65N Group, RX651 Group User’s Manual: Hardware”.

(1) Overview of the TPC on the H8/3048 Device

The basic function of the TPC is to output pulses from pins TP0 to TP15 by using a 16-bit integrated timer unit (ITU) as the time base. The 16-bit TPC consists of four 4-bit output groups (0 to 3). These groups can operate concurrently or independently.

Figure 2.9 shows the block diagram of the TPC.

![Block Diagram of the TPC on the H8/3048 Device](image)
(2) Example settings for achieving a TPC-equivalent function on the RX231 device

Figure 2.10 shows an example of the settings for achieving a TPC-equivalent function on the RX231 device. In this example, general-purpose ports, the DMAC, and the CMT are used to achieve the function.

Description of the substitutive modules

[1] Substitute for the TPC time base on the H8/3048 device
   The CMT is used as the time base and a compare match interrupt of the CMT is used as a trigger to activate the DMAC.

[2] Substitute for the TPC NDRA/NDRB on the H8/3048 device
   [2-1]: A variable that stores the output pattern is prepared in the memory.
   [2-2]: Upon receiving the compare match, the interrupt controller activates the DMAC.
       The DMAC transfers the content of the variable (output pattern) to Pmn. PODR.

[3] Substitute for the TPC output pins (TP0 to TP15) on the H8/3048 device
   Pmn is assigned for general-purpose output so that it can be used for pulse output.

[4] Substitute for TPC PADR/PBDR on the H8/3048 device
   Use Pmn. PODR as the substitute. Output data is stored to Pmn. PODR and is output from Pmn.
   (m = 0 to 9, A to F, H, J to L; n = 0 to 7)
(3) Description of the operation of the example TPC-equivalent function

Figure 2.11 describes the operation of the TPC on the H8/3048 device and the operation of the example TPC-equivalent function on the RX231 device.

<table>
<thead>
<tr>
<th>No.</th>
<th>Operation of TPC on the H8/3048</th>
<th>Example Settings for Replacing the TPC Functionality on the RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ITU CMT value</td>
<td>CMT CMCNT value</td>
</tr>
<tr>
<td></td>
<td>ITU compare match interrupt</td>
<td>CMT compare match interrupt</td>
</tr>
<tr>
<td></td>
<td>ITU compare match interrupt</td>
<td>CMT compare match interrupt</td>
</tr>
<tr>
<td></td>
<td>ITU compare match interrupt</td>
<td>CMT compare match interrupt</td>
</tr>
<tr>
<td></td>
<td>ITU compare match interrupt</td>
<td>CMT compare match interrupt</td>
</tr>
<tr>
<td></td>
<td>ITU compare match interrupt</td>
<td>CMT compare match interrupt</td>
</tr>
<tr>
<td>2</td>
<td>CMT CMT value</td>
<td>CMT CMT value</td>
</tr>
<tr>
<td></td>
<td>ITU CMT value</td>
<td>CMT CMT value</td>
</tr>
<tr>
<td></td>
<td>ITU CMT value</td>
<td>CMT CMT value</td>
</tr>
<tr>
<td></td>
<td>ITU CMT value</td>
<td>CMT CMT value</td>
</tr>
<tr>
<td></td>
<td>ITU CMT value</td>
<td>CMT CMT value</td>
</tr>
<tr>
<td></td>
<td>ITU CMT value</td>
<td>CMT CMT value</td>
</tr>
<tr>
<td>3</td>
<td>TPC NDR/NDRB</td>
<td>DMAC and general-purpose port</td>
</tr>
<tr>
<td></td>
<td>PADR/PBDR</td>
<td>Data transfer by DMAC</td>
</tr>
<tr>
<td></td>
<td>TP0 to TP15</td>
<td>Data transfer by DMAC</td>
</tr>
<tr>
<td></td>
<td>m n</td>
<td>Data transfer by DMAC</td>
</tr>
<tr>
<td></td>
<td>m n</td>
<td>Data transfer by DMAC</td>
</tr>
<tr>
<td></td>
<td>m n</td>
<td>Data transfer by DMAC</td>
</tr>
</tbody>
</table>

Note: 1. “m” = 0 to 5, A to E, H, or J (port number); “n” = 0 to 7

Figure 2.11 Description of the Operation of the Example TPC-Equivalent Function
2.9 Watchdog Timers

2.9.1 Comparison of Specifications

The H8/3048 device has a watchdog timer module called “WDT”. The RX231 device has two watchdog timer modules, WDTA and IWDTa. IWDTa operates based on an independent dedicated clock. Table 2.22 compares the specifications of these modules.

Table 2.22 Comparison of the Specifications of the Watchdog Timer Modules on the H8/3048 and RX231 Devices (WDT, WDTA, and IWDTa)

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
<th>IWDTa</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count source</td>
<td>System clock</td>
<td>Peripheral module clock B (PCLKB)</td>
<td>IWDTCLK (IWDT-dedicated clock)</td>
</tr>
<tr>
<td>Clock division ratio</td>
<td>φ/2, φ/32, φ/64, φ/128, φ/256, φ/512, φ/2048, or φ/4096</td>
<td>1/4, 1/64, 1/128, 1/512, 1/2048, or 1/8192</td>
<td>1/1, 1/16, 1/32, 1/64, 1/128, or 1/256</td>
</tr>
<tr>
<td>Behavior of counting</td>
<td>Up-counting using an 8-bit up-counter</td>
<td>Down-counting using a 14-bit down-counter</td>
<td></td>
</tr>
<tr>
<td>Interrupt source</td>
<td>• Overflow of the up-counter</td>
<td>• Underflow of the down-counter • Refresh performed outside the period in which a refresh is permitted (refresh error)</td>
<td></td>
</tr>
<tr>
<td>Behavior on underflow</td>
<td>—</td>
<td>Reset output or NMI interrupt</td>
<td></td>
</tr>
<tr>
<td>Other items</td>
<td>• Use as an interval timer is possible.</td>
<td>• Use as an interval timer is not possible. • Window function • Auto-start mode • Register start mode</td>
<td></td>
</tr>
</tbody>
</table>
### 2.10 A/D Converter

#### 2.10.1 Comparison of Specifications

Table 2.23 compares the specifications of the A/D converters on the H8/3048 and RX231 device.

**Table 2.23 Comparison of Specifications of the A/D Converters on the H8/3048 and RX231 Devices**

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution</td>
<td>10 bits</td>
<td>12 bits</td>
</tr>
<tr>
<td>A/D conversion method</td>
<td>Successive approximation</td>
<td>Successive approximation</td>
</tr>
<tr>
<td>Conversion time</td>
<td>7.45 μs (min.) per channel</td>
<td>0.83 μs per channel</td>
</tr>
<tr>
<td></td>
<td>(during operation at 18 MHz)</td>
<td>(when the A/D conversion clock ADCLK is operating at 54 MHz)</td>
</tr>
<tr>
<td>Operating mode</td>
<td>Single mode</td>
<td>Single-scan mode</td>
</tr>
<tr>
<td></td>
<td>Scan mode</td>
<td>Continuous-scan mode</td>
</tr>
<tr>
<td></td>
<td>—</td>
<td>Group-scan mode with group-A priority control enabled or disabled</td>
</tr>
<tr>
<td>A/D conversion start condition</td>
<td>• Software</td>
<td>• Software trigger</td>
</tr>
<tr>
<td></td>
<td>• External trigger signal</td>
<td>• Synchronous trigger (trigger signal from MTU, ELC, or TPU)</td>
</tr>
<tr>
<td></td>
<td>(_ADTRG)</td>
<td>• Asynchronous trigger (ADTRG0# pin)</td>
</tr>
<tr>
<td>Other functions</td>
<td>Sample-and-hold function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Variable number of sampling states</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Self-diagnosis for 12-bit A/D converter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Addition mode or averaging mode can be selected for A/D conversion values</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Analog input disconnection detection assist function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Double trigger mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• A/D data register automatic clearing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Compare (windows A and B)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• 16 ring buffers available when the compare function is in use</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Event link function</td>
<td></td>
</tr>
<tr>
<td>Interrupt source</td>
<td>(A/D conversion end interrupt request)</td>
<td>• The scan end interrupt request (S12ADI0) is generated when a scan ends in modes other than double trigger mode and group-scan mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• In double trigger mode, the scan end interrupt request (S12ADI0) is generated when two scans end.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• In group-scan mode, the scan end interrupt request (S12ADI0) is generated when a scan for group A ends. When a scan for group B ends, GBADI, which is the scan end interrupt request dedicated to group B, is generated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• In group-scan mode with double trigger mode selected, the scan end interrupt request (S12ADI0) is generated when two scans for group A end. When a scan for group B ends, GBADI, which is the scan end interrupt request dedicated to group B, is generated.</td>
</tr>
<tr>
<td>Low power consumption</td>
<td>The modules can be placed in standby state.</td>
<td>The modules can be placed in stopped state.</td>
</tr>
<tr>
<td>Conversion target</td>
<td>Analog input pins 0 to 7</td>
<td>Analog input pins 0 to 7 and 16 to 31</td>
</tr>
<tr>
<td></td>
<td>Internal reference voltage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Temperature sensor</td>
<td></td>
</tr>
</tbody>
</table>
### 2.10.2 Operating Modes

The A/D converter on the H8/3048 device can operate in two modes: single mode and scan mode. Table 2.24 shows the correspondence between the conversion modes of the H8/3048 and RX231 devices.

#### Table 2.24 Correspondence of A/D Converter Operating Modes

<table>
<thead>
<tr>
<th>No.</th>
<th>H8/3048</th>
<th>RX210</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Single mode</td>
<td>Single-scan mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(conversion on only one channel)</td>
</tr>
<tr>
<td>2</td>
<td>Scan mode</td>
<td>Continuous-scan mode</td>
</tr>
<tr>
<td>3</td>
<td>—</td>
<td>Group-scan mode</td>
</tr>
</tbody>
</table>

The following table provides an overview of the operating modes.

#### Table 2.25 Overview of the A/D Converter Operating Modes

<table>
<thead>
<tr>
<th>Microcontroller</th>
<th>Operating Mode</th>
<th>Operational Overview</th>
</tr>
</thead>
</table>
| H8/3048         | Single mode    | (1) A/D conversion is performed once on the single specified channel only.  
|                 |                | (2) If interrupts are enabled, an ADI interrupt is generated.  
|                 |                | (3) The value of the ADST bit in ADCSR is held at 1 (A/D converter started) while A/D conversion is in progress and automatically cleared to 0 when A/D conversion finishes, ending A/D converter operation. |
|                 | Scan mode      | (1) Conversion is performed successively on analog input from the specified channels (or channel), starting from the lowest numbered channel.  
|                 |                | (2) When conversion of all the specified channels finishes, an ADI interrupt is generated.  
|                 |                | (3) Steps 1 and 2 are repeated for as long as the ADST bit in ADCSR remains set to 1 (A/D converter started). A/D conversion ends when the ADST bit is cleared to 0. |
| RX231           | Single-scan mode | (1) Conversion is performed successively on analog input from the specified channels (or channel), starting from the lowest numbered channel.  
|                 |                | (2) When conversion of all the specified channels finishes, an S12ADI0 interrupt is generated.  
|                 |                | (3) The value of the ADST bit in ADCSR is held at 1 (start of A/D conversion) while A/D conversion is in progress, and is automatically cleared to 0 when A/D conversion of the selected channels is completed. |
|                 | Continuous-scan mode | The operation in single-scan mode shown above is repeated multiple times. |
|                 | Group-scan mode | (1) Scanning of group A and group B starts when the selected trigger corresponding to the group occurs, the selected channels in group A and group B are scanned once each, and operation ends.  
|                 |                | (2) If interrupts are enabled for group A, an S12ADI0 interrupt is generated when scanning of group A finishes. If interrupts are enabled for group B, a GBADI interrupt is generated when scanning of group B finishes. |
### 2.10.3 List of Registers

Table 2.26 compares the functions of the registers for the A/D converter on the H8/3048 device and the 12-bit A/D converter on the RX231 device.

<table>
<thead>
<tr>
<th>Register Type</th>
<th>H8/3048</th>
<th>RX231</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>A/D data registers A to D</td>
<td>A/D data registers A to D (ADDRA to ADDRD)</td>
<td>A/D data register y (ADDBy) (y = 0 to 7 or 16 to 31)</td>
<td>Present*1</td>
</tr>
<tr>
<td>A/D control/status register</td>
<td>A/D control/status register (ADCSR)</td>
<td>A/D control register (ADCSR)</td>
<td>Present</td>
</tr>
<tr>
<td>A/D control register</td>
<td>A/D control register (ADCSR)</td>
<td>A/D control register (ADCSR)</td>
<td>Present</td>
</tr>
<tr>
<td>A/D channel select register A0</td>
<td>A/D channel select register A0 (ADANSA0)</td>
<td>A/D channel select register A0 (ADANSA0)</td>
<td></td>
</tr>
<tr>
<td>A/D channel select register A1</td>
<td>A/D channel select register A1 (ADANSA1)</td>
<td>A/D channel select register A1 (ADANSA1)</td>
<td></td>
</tr>
<tr>
<td>A/D channel select register B0</td>
<td>A/D channel select register B0 (ADANSB0)</td>
<td>A/D channel select register B0 (ADANSB0)</td>
<td></td>
</tr>
<tr>
<td>A/D channel select register B1</td>
<td>A/D channel select register B1 (ADANSB1)</td>
<td>A/D channel select register B1 (ADANSB1)</td>
<td></td>
</tr>
<tr>
<td>A/D sampling state register</td>
<td>A/D sampling state register n (ADSSTRn) (n = 0 to 7, L, T, or O)</td>
<td>A/D sampling state register n (ADSSTRn) (n = 0 to 7, L, T, or O)</td>
<td></td>
</tr>
<tr>
<td>A/D conversion start trigger</td>
<td>A/D conversion start trigger select register (ADSTRGR)</td>
<td>Present</td>
<td></td>
</tr>
<tr>
<td>A/D data duplication register</td>
<td>A/D data duplication register (ADDBLDR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D temperature sensor data</td>
<td>A/D temperature sensor data register (ADTSDR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D internal reference voltage</td>
<td>A/D internal reference voltage data register (ADOCVR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D self-diagnosis data register</td>
<td>A/D self-diagnosis data register (ADSDSR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D-converted value addition/averaging function channel select register</td>
<td>A/D-converted value addition/averaging function channel select register n (ADADSn) (n = 0 or 1)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D-converted value addition/averaging count select register</td>
<td>A/D-converted value addition/averaging count select register (ADADC)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D control extended register</td>
<td>A/D control extended register (ADCEER)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D conversion extended input control register</td>
<td>A/D conversion extended input control register (ADEICR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D disconnection detection control register</td>
<td>A/D disconnection detection control register (ADDISCR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D sampling state register</td>
<td>A/D sampling state register n (ADSSTRn) (n = 0 to 7, L, T, or O)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D disconnection detection control register</td>
<td>A/D disconnection detection control register (ADDISCR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D event linkage control register</td>
<td>A/D event linkage control register (ADELCR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D group-scan priority control</td>
<td>A/D group-scan priority control register (ADGSPCR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D compare function control</td>
<td>A/D compare function control register (ADCMPCR)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-A</td>
<td>A/D compare function window-A channel select register n (ADCMPANSRn) (n = 0 or 1)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-A</td>
<td>A/D compare function window-A extended input select register (ADCMPANSER)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-A</td>
<td>A/D compare function window-A comparison condition setting register n (ADCMPPLRn) (n = 0 or 1)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
</tbody>
</table>
## Table 2.26  Comparison of the Functions of the Registers for the AD Converters (2/2)

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>A/D converter</td>
<td>12-bit A/D converter</td>
<td>— (There is no corresponding register.)</td>
</tr>
<tr>
<td>A/D compare function window-A extended input comparison condition setting register (ADCMPLER)</td>
<td>New register added on the RX231</td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-A lower-bit level setting register (ADCMPDR0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-A upper-bit level setting register (ADCMPDR1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-A channel status register n (ADCMPSRn) (n = 0 or 1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-A extended input channel status register (ADCMPSER)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D high-potential/low-potential reference voltage control register (ADHVREFCNT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-A/B status monitor register (ADWINMON)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-B channel select register (ADCMPBNSR)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-B lower-bit level setting register (ADWINLLB)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-B upper-bit level setting register (ADWINULB)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D compare function window-B channel status register (ADCMPBSR)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D data storage buffer register n (ADBUFn) (n = 0 to 15)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D data storage buffer enable register (ADBUFEN)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D data storage buffer pointer register (ADBUFPTR)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: 1. On the H8/3048 and RX231 devices, these registers are 16-bit registers. Note that, on the H8/3048 device, the 10 active bits are left-aligned, whereas on the RX231 device, the 12 active bits are right-aligned in the initial (default) state. (Left-aligned format can be selected by setting the ADRFMT bit in ADCER to 1.)

### 2.10.4 Module Stop

On the RX231 device, after recovery from a reset, modules other than the DMAC, DTC, and RAM are placed in stopped state (clock supply is stopped) to reduce power consumption. Before configuring the modules, make sure that they are not in stopped state.
2.11 D/A Converter

2.11.1 Comparison of Specifications

Table 2.27 Comparison of Specifications of the D/A Converters on the H8/3048 and RX231 Devices

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>D/A converter</td>
<td>8 bits</td>
<td>12-bit D/A converter (R12DA)</td>
</tr>
<tr>
<td>Resolution</td>
<td></td>
<td>12 bits</td>
</tr>
<tr>
<td>Number of input channels</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Conversion speed</td>
<td>10 μs (at a load capacitance of 20 pF)</td>
<td>30 μs, max.</td>
</tr>
<tr>
<td>Output voltage</td>
<td>0 V to 255/256 × V&lt;sub&gt;REF&lt;/sub&gt;</td>
<td>0 V to 4095/4096 × V&lt;sub&gt;REF&lt;/sub&gt;*1</td>
</tr>
<tr>
<td>D/A output hold function in software standby mode</td>
<td>Implemented</td>
<td>Implemented</td>
</tr>
<tr>
<td>Low power consumption</td>
<td>—</td>
<td>The modules can be placed in stopped state.*2</td>
</tr>
</tbody>
</table>

Notes:
1. Voltage in the range from 0.35 V to (AVCC0 - 0.47) V can be output. One of the following can be selected: AVCC0/AVSS, Internal reference voltage/AVSS, or VREFH/VREFL.
2. Do not place the 12-bit A/D converter module in stopped state if the DAADST bit in the DAADSCR register is 1 (measure against interference between D/A conversion and A/D conversion is enabled). In this case, stopping the A/D converter may also cause the D/A converter to stop.

2.11.2 Comparison of Registers

This section shows the differences in the registers between the H8/3048 and RX231 devices.

Table 2.28 Comparison of Registers on the H8/3048 and RX231 Devices

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>D/A data registers 0 and 1 (DADR0 and DADR1)</td>
<td>D/A data registers 0 and 1 (DADR0 and DADR1)</td>
<td>Present*1</td>
</tr>
<tr>
<td>D/A control register (DACR)</td>
<td>D/A control register (DACR)</td>
<td>Present*1</td>
</tr>
<tr>
<td>D/A standby control register (DASTCR)</td>
<td>—*2</td>
<td>—</td>
</tr>
<tr>
<td>DADRm format select register (DADPR)</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>D/A and A/D conversions synchronization start control register (DAADSCR)</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>D/A VREF control register (DAVREFCR)</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>

Notes:
1. Bit assignments are different.
2. The RX231 device does not have a register equivalent to the D/A standby control register (DASTCR) of the H8/3048 device.

2.11.3 Module Stop

On the RX231 device, after recovery from a reset, modules other than the DMAC, DTC, and RAM are placed in stopped state (clock supply is stopped) to reduce power consumption. Before configuring the modules, make sure that they are not in stopped state.
2.12 Flash Memory

2.12.1 Comparison of Specifications

Table 2.29  Comparison of Specifications of Flash Memory on the H8/3048F and RX231 Devices

<table>
<thead>
<tr>
<th>Item</th>
<th>H8/3048F</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>128 KB</td>
<td>User area: 512 KB, max.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data area: 8 KB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extra area: Size enough</td>
</tr>
<tr>
<td></td>
<td></td>
<td>to store the startup</td>
</tr>
<tr>
<td></td>
<td></td>
<td>area information, access</td>
</tr>
<tr>
<td></td>
<td></td>
<td>window information, and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>unique IDs</td>
</tr>
<tr>
<td>Block size × Number</td>
<td>• 16 KB × 7 (112 KB)</td>
<td>• User area: 2 KB × 256</td>
</tr>
<tr>
<td>of blocks</td>
<td>• 12 KB × 1 (12 KB)</td>
<td>(512 KB)</td>
</tr>
<tr>
<td></td>
<td>• 512 bytes × 8 (4 KB)</td>
<td>• Data area: 1 KB × 8</td>
</tr>
<tr>
<td>Operating mode</td>
<td>• Program mode</td>
<td>(8 KB)</td>
</tr>
<tr>
<td></td>
<td>• Program-and-verify mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Erase mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Erase-and-verify mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Pre-write-and-verify</td>
<td></td>
</tr>
<tr>
<td>ROM</td>
<td></td>
<td>E2 data flash access</td>
</tr>
<tr>
<td>Unit of data to</td>
<td>Write: In bytes</td>
<td>prohibition mode</td>
</tr>
<tr>
<td>write/erase</td>
<td>Erase: In blocks</td>
<td>ROM/E2 data flash read</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ROM program/erase mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E2 data flash program/erase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mode</td>
</tr>
<tr>
<td>Write/erase time</td>
<td>System clock: 1 to 16 MHz</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Write: 50 μs or 1000 μs</td>
<td>FCLK: 32 MHz</td>
</tr>
<tr>
<td></td>
<td>(per byte)*1</td>
<td>• Write: 52.3 μs or 491 μs</td>
</tr>
<tr>
<td></td>
<td>• Erase: 1 s or 30 s</td>
<td>(per 8 bytes)*1</td>
</tr>
<tr>
<td></td>
<td>(per 128 KB (entire block))</td>
<td>• Erase: 66.7 ms or 1469 ms</td>
</tr>
<tr>
<td></td>
<td>*1</td>
<td>(per 512 KB (entire block))</td>
</tr>
<tr>
<td>Maximum rewrite count</td>
<td>100</td>
<td>1000</td>
</tr>
<tr>
<td>E2 data flash</td>
<td>—</td>
<td>Write: In bytes</td>
</tr>
<tr>
<td>Unit of data to</td>
<td>—</td>
<td>Erase: In blocks</td>
</tr>
<tr>
<td>write/erase</td>
<td>—</td>
<td>FCLK: 32 MHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Write: 40.8 μs or 376 μs</td>
</tr>
<tr>
<td>Write/erase time</td>
<td>—</td>
<td>(per byte)*1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Erase: 12.9 ms or 368 ms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(per 8 KB (entire block)) *1</td>
</tr>
<tr>
<td>Maximum rewrite count</td>
<td>—</td>
<td>1,000,000 / 100,000*2</td>
</tr>
<tr>
<td>Programming mode</td>
<td>• On-board programming</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Boot mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>User program mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PROM mode</td>
<td></td>
</tr>
<tr>
<td>Other items</td>
<td>Automatic bit rate</td>
<td>Automatic bit rate</td>
</tr>
<tr>
<td></td>
<td>adjustment</td>
<td>adjustment</td>
</tr>
<tr>
<td></td>
<td>Protect mode</td>
<td>Protection</td>
</tr>
</tbody>
</table>

Notes: 1. The values before and after a slash (/) respectively indicate the typical and maximum values.
2. The values before and after a slash (/) respectively indicate the typical and minimum values.
2.13 Low-Power Modes and Module Stop Function

2.13.1 Comparison of Low-Power Mode Specifications

The H8/3048 device provides three low-power modes. The RX231 device also provides three low-power modes. On the H8/3048 and RX231 devices, in a low-power mode, the CPU stops and the operating states of the internal modules change according to the selected low-power mode. The following table describes the low-power modes.

(1) Low-power modes of the H8/3048 device

Table 2.30 shows the operating states of the internal modules in each low-power mode of the H8/3048 device.

Table 2.30 Operating States of Internal Modules in Low-Power Modes of the H8/3048 Device

<table>
<thead>
<tr>
<th>Module</th>
<th>Sleep mode</th>
<th>Software standby mode</th>
<th>Hardware standby mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock oscillator</td>
<td>Operating</td>
<td>Stopped</td>
<td>Stopped</td>
</tr>
<tr>
<td>CPU</td>
<td>Stopped</td>
<td>Stopped</td>
<td>Stopped</td>
</tr>
<tr>
<td>Registers</td>
<td>Stopped/Reset</td>
<td>Settings retained</td>
<td>Settings undetermined</td>
</tr>
<tr>
<td>Refresh controller</td>
<td>Operating</td>
<td>Stopped/Reset</td>
<td>Stopped/Reset</td>
</tr>
<tr>
<td>Other peripherals</td>
<td>Operating</td>
<td>Stopped/Reset</td>
<td>Stopped/Reset</td>
</tr>
<tr>
<td>RAM</td>
<td>Settings retained</td>
<td>Settings retained</td>
<td>Settings retained</td>
</tr>
<tr>
<td>φ clock output</td>
<td>φ output</td>
<td>High-level output</td>
<td>High-impedance</td>
</tr>
<tr>
<td>I/O ports</td>
<td>Settings retained</td>
<td>Settings retained</td>
<td>High-impedance</td>
</tr>
</tbody>
</table>

Note: 1. Bits 7 and 6 in the RTCNT and RTMCSR registers are reset.
**Low-power modes of the RX231 device**

Table 2.31 shows the operating states of the internal modules in each low-power mode of the RX231 device.

<table>
<thead>
<tr>
<th>Module</th>
<th>Sleep mode</th>
<th>Deep sleep mode</th>
<th>Software standby mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main clock oscillator</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Stopped</td>
</tr>
<tr>
<td>Subclock oscillator</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
</tr>
<tr>
<td>High-speed on-chip oscillator</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Stopped</td>
</tr>
<tr>
<td>Low-speed on-chip oscillator</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
</tr>
<tr>
<td>IWDT-dedicated on-chip oscillator</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
</tr>
<tr>
<td>PLL</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Stopped</td>
</tr>
<tr>
<td>USB-dedicated PLL</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Stopped</td>
</tr>
<tr>
<td>CPU</td>
<td>Stopped (Settings retained)</td>
<td>Stopped (Settings retained)</td>
<td>Stopped (Settings retained)</td>
</tr>
<tr>
<td>RAM (0000 0000h to 0000 FFFFh)</td>
<td>Possible to operate (Settings retained)</td>
<td>Stopped (Settings retained)</td>
<td>Stopped (Settings retained)</td>
</tr>
<tr>
<td>Flash memory</td>
<td>Operating</td>
<td>Stopped (Settings retained)</td>
<td>Stopped (Settings retained)</td>
</tr>
<tr>
<td>Watchdog timer (WDT)</td>
<td>Stopped (Settings retained)</td>
<td>Stopped (Settings retained)</td>
<td>Stopped (Settings retained)</td>
</tr>
<tr>
<td>Independent watchdog timer (IWDT)</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
</tr>
<tr>
<td>Realtime clock (RTC)</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
</tr>
<tr>
<td>Low-power timer (LPT)</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
</tr>
<tr>
<td>Voltage detection circuit (LVD)</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
</tr>
<tr>
<td>Power-on reset circuit</td>
<td>Operating</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>Peripheral modules</td>
<td>Possible to operate</td>
<td>Possible to operate</td>
<td>Stopped (Settings retained)</td>
</tr>
<tr>
<td>I/O ports</td>
<td>Operating</td>
<td>Operating</td>
<td>Settings retained</td>
</tr>
</tbody>
</table>

**Note:**
- “Possible to operate” indicates that the operating state (operating or stopped) can be controlled by configuring control registers.
- “Stopped (Settings retained)” indicates that the module is stopped and the settings of the internal registers are retained.
2.13.2 Mode Transitions

Figure 2.12 shows the transitions of low-power modes on the RX231 device.

![Mode Transitions Diagram]

**Notes:**
1. When the WAIT instruction is executed, if an interrupt that triggers cancellation is accepted during state transition to the program halt state, the transition is canceled and interrupt exception handling is performed.
2. External pin interrupt (NMI or IRQ0 to IRQ7), peripheral function interrupt (RTC alarm, RTC cycle, IWDT, voltage monitoring, detection of VBATT pin voltage reduction, USB, or ELC (LPT-dedicated interrupt))
3. Transition from the RESET state becomes the LOCO clock source in normal mode.
4. The mode changes from sleep mode, deep sleep mode, or software standby mode to the normal operating mode upon an interrupt.
   - If the mode changes from sleep mode, the clock source to be used after sleep mode ends can be selected.
   - For details, refer to the description of the RSTCLKCR register.
   - If the mode changes from deep sleep mode or software standby mode, the clock source does not change when the mode changes.

**Note:** The state always changes to reset state if a RES# pin reset, power-on reset, voltage monitoring reset, WDT reset, IWDT reset, or software reset occurs.

*Figure 2.12  Transitions of Modes on the RX231 Device*
2.13.3 Module Stop
The H8/3048 device provides the module standby function, which can stop the peripheral modules independently of the low-power mode. When the operating state changes to the module standby state, the states of the internal modules are initialized. On the H8/3048 device, after a reset is performed, the peripheral modules are operating. The RX231 device also provides a similar function, the module stop function, which can stop the peripheral modules independently of the low-power mode. When the operating state changes to the module stop state, the states of the internal modules are retained. On the RX231 device, after recovery from a reset, all peripheral modules except the DMAC, DTC, and RAM are stopped. To start the peripheral modules, cancel module stop. The operating state (operating or stopped) of each peripheral module is controlled by configuring the registers shown in Table 2.32.

Table 2.32 List of Modules That Can Be Stopped

<table>
<thead>
<tr>
<th>H8/3048</th>
<th>RX231</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module standby function</td>
<td>Module stop function</td>
</tr>
<tr>
<td>MSTCR: Module standby control register</td>
<td>SYSTEM.MSTPCRA: Module stop control register A</td>
</tr>
<tr>
<td>ITU</td>
<td>TMR</td>
</tr>
<tr>
<td>SCI</td>
<td>MTU</td>
</tr>
<tr>
<td>DMA</td>
<td>TPU</td>
</tr>
<tr>
<td>Refresh controller</td>
<td>CMT</td>
</tr>
<tr>
<td>A/D converter</td>
<td>S12AD</td>
</tr>
<tr>
<td>—</td>
<td>DA</td>
</tr>
<tr>
<td>—</td>
<td>DMAC and DTC</td>
</tr>
<tr>
<td>SYSTEM.MSTPCRB: Module stop control register B</td>
<td></td>
</tr>
<tr>
<td>RSCAN0</td>
<td></td>
</tr>
<tr>
<td>SCI0, SCI1, SCI5, SCI6, and SCI12</td>
<td></td>
</tr>
<tr>
<td>DOC</td>
<td></td>
</tr>
<tr>
<td>ELC</td>
<td></td>
</tr>
<tr>
<td>Comparator B</td>
<td></td>
</tr>
<tr>
<td>RSPI0</td>
<td></td>
</tr>
<tr>
<td>USB0</td>
<td></td>
</tr>
<tr>
<td>RIC0</td>
<td></td>
</tr>
<tr>
<td>CRC</td>
<td></td>
</tr>
<tr>
<td>SYSTEM.MSTPCRC: Module stop control register C</td>
<td></td>
</tr>
<tr>
<td>RAM</td>
<td></td>
</tr>
<tr>
<td>CAC</td>
<td></td>
</tr>
<tr>
<td>IRDA</td>
<td></td>
</tr>
<tr>
<td>SCI8 and SCI9</td>
<td></td>
</tr>
<tr>
<td>SYSTEM.MSTPCRD: Module stop control register D</td>
<td></td>
</tr>
<tr>
<td>CTSU</td>
<td></td>
</tr>
<tr>
<td>SSI</td>
<td></td>
</tr>
<tr>
<td>SDHI</td>
<td></td>
</tr>
<tr>
<td>Trusted Secure IP</td>
<td></td>
</tr>
</tbody>
</table>
3. Reference Documents

This chapter lists the documents referenced in the preparation of this application note. When referring to the documents listed below, substitute the latest version if a newer version is available. The latest versions of these documents can be confirmed and downloaded from the Renesas Electronics website.

Table 3.1 Reference Documents

<table>
<thead>
<tr>
<th>Reference Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>H8/3048 Group, H8/3048 F-ZTAT™ Hardware Manual (REJ09B0259)</td>
</tr>
<tr>
<td>RX230 Group, RX231 Group User’s Manual: Hardware (R01UH0496EJ)</td>
</tr>
<tr>
<td>RX231 Group Initial Setting (R01AN2185EJ)</td>
</tr>
<tr>
<td>RX65N Group, RX651 Group User’s Manual: Hardware (R01UH0590EJ)</td>
</tr>
</tbody>
</table>
## Revision History

<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Mar. 27, 2023</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>First edition issued</td>
</tr>
</tbody>
</table>
General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit 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. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can 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 must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

2. Processing at power-on
   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.

3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. 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. Follow the guideline for input signal during power-off state as described in your product documentation.

4. Handling of unused pins
   Handle unused pins in accordance 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 the 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.

5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.

6. 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 $V_{IL}$ (Max.) and $V_{IH}$ (Min.) due to noise, for example, 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 $V_{IL}$ (Max.) and $V_{IH}$ (Min.).

7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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.
Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of circuits, software, or information.

2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.

3. A license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.

4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.

5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.

6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depend on the product’s quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.

7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to, unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACH, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.

8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Microcontroller Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.

9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.

10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.

11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.

12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.

13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.

14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Keto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

© 2023 Renesas Electronics Corporation. All rights reserved.