Introduction

This application note describes how to change the RL78/I1D's CPU clock and set it to standby (changing operation modes).

This application uses switch input to change the CPU clock and the operation mode, while controlling 5 LEDs to indicate the CPU clock status and the operation mode.

Target Device

RL78/I1D

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.
## Contents

1. Specifications .................................................................................................................... 4  
   1.1 CPU Clock Changes ........................................................................................................ 7  
   1.1.1 Changing from high-speed on-chip oscillator clock to middle-speed on-chip oscillator clock ................................................................. 8  
   1.1.2 Changing from high-speed on-chip oscillator clock to low-speed on-chip oscillator clock ................................................................. 9  
   1.1.3 Changing from high-speed on-chip oscillator clock to high-speed system clock ................................................................. 10  
   1.1.4 Changing from high-speed on-chip oscillator clock to sub clock ................................................................................................. 12  
   1.1.5 Changing from middle-speed on-chip oscillator clock to high-speed on-chip oscillator clock ................................................................. 13  
   1.1.6 Changing from middle-speed on-chip oscillator clock to low-speed on-chip oscillator clock ................................................................. 14  
   1.1.7 Changing from middle-speed on-chip oscillator clock to high-speed system clock ................................................................. 15  
   1.1.8 Changing from middle-speed on-chip oscillator clock to sub clock ................................................................................................. 17  
   1.1.9 Changing from low-speed on-chip oscillator clock to high-speed on-chip oscillator clock ................................................................. 18  
   1.1.10 Changing from low-speed on-chip oscillator clock to middle-speed on-chip oscillator clock ................................................................. 19  
   1.1.11 Changing from low-speed on-chip oscillator clock to high-speed system clock ................................................................. 20  
   1.1.12 Changing from high-speed system clock to high-speed on-chip oscillator clock ................................................................. 22  
   1.1.13 Changing from high-speed system clock to middle-speed on-chip oscillator clock ................................................................. 23  
   1.1.14 Changing from high-speed system clock to low-speed on-chip oscillator clock ................................................................. 24  
   1.1.15 Changing from high-speed system clock to sub clock ................................................................................................. 25  
   1.1.16 Changing from sub clock to high-speed on-chip oscillator clock ................................................................................................. 26  
   1.1.17 Changing from sub clock to middle-speed on-chip oscillator clock ................................................................................................. 27  
   1.1.18 Changing from sub clock to high-speed system clock ................................................................................................. 28  

2. Operation Confirmation Conditions .................................................................................................................... 29

3. Related Application Notes .................................................................................................................... 29

4. Hardware Explanation .................................................................................................................... 30
   4.1 Hardware Configuration Example ................................................................................................. 30
   4.2 Used Pin List ................................................................................................................................. 30

5. Software Explanation .................................................................................................................... 31
   5.1 Operation Outline ................................................................................................................................. 31
   5.2 Option Byte Settings ................................................................................................................................. 34
   5.3 Variables .................................................................................................................................................. 34
   5.4 Functions (subroutines) ................................................................................................................................. 35
   5.5 Function (subroutine) Specifications ................................................................................................................................. 37
   5.6 Flowcharts .................................................................................................................................................. 47
   5.6.1 Main Processing .................................................................................................................................................. 48
   5.6.2 Initialization Function .................................................................................................................................................. 52
   5.6.3 System Function .................................................................................................................................................. 53
   5.6.4 Input/Output Port Settings ................................................................................................................................. 54
   5.6.5 Clock Generator Setting .................................................................................................................................................. 55
   5.6.6 External Interrupt Setting .................................................................................................................................................. 60
   5.6.7 12-bit Interval Timer Setting .................................................................................................................................................. 62
   5.6.8 Main initializes settings .................................................................................................................................................. 65
   5.6.9 Status Transition AtoB .................................................................................................................................................. 66
   5.6.10 CPU operation (NOP instruction execution) .................................................................................................................................................. 66
   5.6.11 Status Transition BtoE .................................................................................................................................................. 67
   5.6.12 Error Processing of Status Transition .................................................................................................................................................. 69
   5.6.13 Status Transition EtoO .................................................................................................................................................. 69
   5.6.14 Status Transition OtoE .................................................................................................................................................. 70
   5.6.15 Status Transition EtoB .................................................................................................................................................. 71
   5.6.16 Status Transition BtoD .................................................................................................................................................. 73
   5.6.17 Status Transition DtoE .................................................................................................................................................. 75
   5.6.18 Status Transition EtoD .................................................................................................................................................. 77
   5.6.19 Status Transition DtoM .................................................................................................................................................. 79
   5.6.20 Status Transition MtoD .................................................................................................................................................. 79
5.6.21 Status Transition DtoN ................................................................. 80
5.6.22 Status Transition KtoD ................................................................. 80
5.6.23 Status Transition DtoB ................................................................. 81
5.6.24 Status Transition BtoI ................................................................. 83
5.6.25 Status Transition ItoB ................................................................. 83
5.6.26 Status Transition BtoG ................................................................. 84
5.6.27 Status Transition GtoB ................................................................. 84
5.6.28 Status Transition BtoH ................................................................. 85
5.6.29 A/D Converter Setting ............................................................... 86
5.6.30 A/D Converter Initial Setting ..................................................... 87
5.6.31 Status Transition HtoB ................................................................. 92
5.6.32 Status Transition BtoC ................................................................. 93
5.6.33 Status Transition CtoD ................................................................. 95
5.6.34 Status Transition DtoF ................................................................. 97
5.6.35 Status Transition FtoD ................................................................. 99
5.6.36 Status Transition DtoC ................................................................. 101
5.6.37 Status Transition CtoJ ................................................................. 103
5.6.38 Status Transition JtoC ................................................................. 103
5.6.39 Status Transition CtoK ................................................................. 104
5.6.40 Status Transition KtoC ................................................................. 104
5.6.41 Status Transition CtoL ................................................................. 105
5.6.42 Status Transition LtoC ................................................................. 106
5.6.43 Status Transition CtoE ................................................................. 107
5.6.44 Status Transition EtoC ................................................................. 109
5.6.45 Status Transition CtoF ................................................................. 111
5.6.46 Status Transition FtoC ................................................................. 113
5.6.47 Status Transition CtoB ................................................................. 115
5.6.48 Status Transition BtoF ................................................................. 117
5.6.49 Status Transition FtoP ................................................................. 119
5.6.50 Status Transition PtoF ................................................................. 119
5.6.51 Status Transition FtoB ................................................................. 120
5.6.52 Status Transition End Processing ............................................. 122
5.6.53 External Interrupt Servicing ..................................................... 123
5.6.54 12-bit Interval Timer Interrupt Servicing ................................. 124
5.6.55 A/D Conversion Completion Interrupt Servicing ..................... 124

6. Sample Code .................................................................................. 125

7. Reference Documents ..................................................................... 125
1. Specifications

This application describes how to switch the CPU clock and operation mode using switch input, as shown in Figure 1.1 Operating Mode Status Transition Diagram.

In addition, the application controls five LEDs to indicate the status of the CPU clock and the operation mode.

The Peripheral Functions and Applications used in this application note, Operating Mode Status Transition Diagram, and Operation Modes and Corresponding LED Status are shown in Table 1.1, Figure 1.1, and Table 1.2, correspondingly.

<table>
<thead>
<tr>
<th>Peripheral Function</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port output</td>
<td>Controls LEDs (LED1-LED5) connected to pins P50, P51, P52, P60, P61.</td>
</tr>
<tr>
<td>External interrupt</td>
<td>Interrupt (INTP0) that detects a pin input edge according to switch input (SW1).</td>
</tr>
<tr>
<td>12-bit interval timer</td>
<td>Interrupt (INTIT) that detects an interval signal from the 12-bit interval timer</td>
</tr>
<tr>
<td>A/D converter</td>
<td>Converts analog signal input level of the P20/ANI13 pin.</td>
</tr>
</tbody>
</table>
Figure 1.1 Operating Mode Status Transition Diagram
Table 1.2  Operation Modes and Corresponding LED Status

<table>
<thead>
<tr>
<th>CPU/Peripheral Hardware Clock (fCLK)</th>
<th>Operation mode</th>
<th>LED Status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Normal operation</td>
<td>LED1</td>
</tr>
<tr>
<td>High-speed on-chip oscillator clock</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>(fIH)</td>
<td>HALT mode</td>
<td>OFF</td>
</tr>
<tr>
<td>SNOOZE mode</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>STOP mode</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>Normal operation</td>
<td>ON</td>
</tr>
<tr>
<td>Middle-speed on-chip oscillator</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>clock (fIM)</td>
<td>HALT mode</td>
<td>OFF</td>
</tr>
<tr>
<td>SNOOZE mode</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>STOP mode</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>Normal operation</td>
<td>ON</td>
</tr>
<tr>
<td>Low-speed on-chip oscillator clock</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>(fIL)</td>
<td>HALT mode</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>Normal operation</td>
<td>ON</td>
</tr>
<tr>
<td>High-speed system clock (fMX)</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>HALT mode</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>STOP mode</td>
<td>OFF</td>
</tr>
<tr>
<td>Sub clock (fSX)</td>
<td>Normal operation</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>HALT mode</td>
<td>OFF</td>
</tr>
</tbody>
</table>

Note: Make sure that the current flowing through 1 pin is 8mA or less. Refer to the Electrical characteristics of the user’s manual of RL78/I1D for the current that can flow to the pin.
1.1  CPU Clock Changes

This section describes the special function register (SFR) settings required for changing the CPU clock.

- Changing from high-speed on-chip oscillator clock to middle-speed on-chip oscillator clock
- Changing from high-speed on-chip oscillator clock to low-speed on-chip oscillator clock
- Changing from high-speed on-chip oscillator clock to high-speed system clock
- Changing from high-speed on-chip oscillator clock to sub clock
- Changing from middle-speed on-chip oscillator clock to high-speed on-chip oscillator clock
- Changing from middle-speed on-chip oscillator clock to low-speed on-chip oscillator clock
- Changing from middle-speed on-chip oscillator clock to high-speed system clock
- Changing from middle-speed on-chip oscillator clock to sub clock
- Changing from low-speed on-chip oscillator clock to high-speed on-chip oscillator clock
- Changing from low-speed on-chip oscillator clock to middle-speed on-chip oscillator clock
- Changing from low-speed on-chip oscillator clock to high-speed system clock
- Changing from high-speed system clock to high-speed on-chip oscillator clock
- Changing from high-speed system clock to middle-speed on-chip oscillator clock
- Changing from high-speed system clock to low-speed on-chip oscillator clock
- Changing from high-speed system clock to sub clock
- Changing from sub clock to high-speed on-chip oscillator clock
- Changing from sub clock to middle-speed on-chip oscillator clock
- Changing from sub clock to high-speed system clock
1.1.1 Changing from high-speed on-chip oscillator clock to middle-speed on-chip oscillator clock

When changing the CPU clock from the high-speed on-chip oscillator clock to the middle-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the middle-speed on-chip oscillator clock to $f_{\text{CLK}}$ using the system clock control register (CKC). Confirm that the status of the main on-chip oscillator clock status has switched to the middle-speed on-chip oscillator clock, and then stop the high-speed on-chip oscillator.

① Set (1) the MIOEN bit of the CSC register, and then start oscillating the middle-speed on-chip oscillator.

\[
\begin{array}{cccccccc}
\text{CSC} & \text{MSTOP} & \text{XTSTOP} & 0 & 0 & 0 & 1 & 0 \\
\end{array}
\]

② Use a software wait to wait for the oscillation of the middle-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 4µs) using the timer function or another function.

③ Set(1) the MCM1 bit of the CKC register to specify the middle-speed on-chip oscillator clock as the main on-chip oscillator clock.

\[
\begin{array}{cccccccc}
\text{CKC} & \text{CLS} & \text{CSS} & \text{MCS} & \text{MCM0} & 0 & 0 & 1 \\
\end{array}
\]

④ Confirm that the MCS1 bit of the CKC register has changed to 1, set (1) the HIOSTOP bit and stop the oscillating the high-speed on-chip oscillator.

\[
\begin{array}{cccccccc}
\text{CSC} & \text{MSTOP} & \text{XTSTOP} & 0 & 0 & 0 & 1 & 0 \\
\end{array}
\]

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
### Changing from high-speed on-chip oscillator clock to low-speed on-chip oscillator clock

When changing the CPU clock from the high-speed on-chip oscillator clock to the low-speed on-chip oscillator clock, start the oscillation using the subsystem clock select register (CKSEL), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the low-speed on-chip oscillator clock to $f_{CLK}$ using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the subsystem clock, and then stop the high-speed on-chip oscillator.

1. Set (1) the SELLOSC bit of the CKSEL register, and then start oscillating the low-speed on-chip oscillator.

   ![CKSEL register](Image)

   - Set (1) the SELLOSC bit of the CKSEL register, and then start oscillating the low-speed on-chip oscillator.

2. Use a software wait to wait for the oscillation of the low-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: $210\mu s$) using the timer function or another function. In this application note is always to operate the low-speed on-chip oscillator. For this reason, it does not perform oscillation stabilization wait of the low-speed on-chip oscillator.

3. Set (1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

   ![CKC register](Image)

   - Set (1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

4. Confirm that the CLS bit of the CKC register has changed to 1, set (1) the HIOSTOP bit and stop the oscillating the high-speed on-chip oscillator.

   ![CSC register](Image)

   - Confirm that the CLS bit of the CKC register has changed to 1, set (1) the HIOSTOP bit and stop the oscillating the high-speed on-chip oscillator.

Register setting values:
- x: unused bit; blank space; unchanged bit;
- -: reserved bits or unassigned bit
1.1.3 Changing from high-speed on-chip oscillator clock to high-speed system clock

When changing the CPU clock from the high-speed on-chip oscillator clock to the high-speed system clock, set the oscillator and start oscillation using the following registers: clock operation mode control register (CMC), oscillation stabilization time select register (OSTS), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the oscillation stabilization time counter status register (OSTC).

After the oscillation stabilizes, set the high-speed system clock to \( f_{CLK} \) using the system clock control register (CKC). Confirm that the status of the main system clock has changed to the high-speed system clock, and then stop the high-speed on-chip oscillator.

1. Set (1) the OSCSEL bit of the CMC register (when \( f_x > 10\text{MHz} \), set (1) the AMPH bit) to operate the X1 oscillator. Set (1) the EXCLK bit and OSCSEL bit when using the external clock.

   ![CMC Register](image)

   **AMPH bit:** clear to 0 when the X1 oscillation clock is 10 MHz or lower.

2. Using the OSTS register, select the oscillation stabilization time of the X1 oscillation circuit. This setting does not have to exist at the time the external clock is used.

   Example: Set the following values for a wait of at least 102μs based on a 10 MHz resonator.

   ![OSTS Register](image)

3. Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator. After the external clock is input to the external clock signal, to clear (0) the MSTOP bit.

   ![CSC Register](image)

4. Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize. External clock is not required oscillation stabilization wait.

   Example: Wait until the bits reach the following values for a wait of at least 102μs based on a 10 MHz resonator.

   ![OSTC Register](image)

Register setting values:
- **x:** unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
⑤ Set(1) the MCM0 bit of the CKC register to specify the high-speed system clock as the main system clock.

<table>
<thead>
<tr>
<th>CKC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CLS</td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</td>
<td>0</td>
<td>0</td>
<td>MCS1</td>
<td>MCM1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

⑥ Confirm that the MCS bit of the CKC register has changed to 1, set (1) the HIOSTOP bit and stop the oscillating the high-speed on-chip oscillator.

<table>
<thead>
<tr>
<th>CSC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>1</td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
When changing the CPU clock from the high-speed on-chip oscillator clock to the sub clock, set the oscillator and start oscillation using the following registers: subsystem clock supply mode control register (OSMC), clock operation mode control register (CMC), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the timer or similar function. After the oscillation stabilizes, set the sub clock to \( f_{CLK} \) using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock has switched to the subsystem clock, and then stop the high-speed on-chip oscillator.

① In this application, a 12-bit interval timer is used to count the oscillation stabilization time for the subsystem clock resonator. Set the WUTMMCK0 to 1 to use the low-speed on-chip oscillator clock as the count clock for the 12-bit interval timer. To run only the real-time clock and 12-bit interval timer on the subsystem clock in the STOP mode or HALT mode (during CPU operation with the subsystem clock), set the RTCLPC bit to 1.

<table>
<thead>
<tr>
<th>OSMC</th>
<th>RTCLPC</th>
<th>WUTMMCK0</th>
<th>x</th>
<th>x</th>
<th>x</th>
<th>x</th>
<th>x</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

② Set(1) the OSCSELS bit of the CMC register to operate the XT1 oscillator. Set (1) the EXCLKS bit and OSCSELS bit when using the external clock.

<table>
<thead>
<tr>
<th>CMC</th>
<th>EXCLK</th>
<th>OSCSEL</th>
<th>EXCLKS</th>
<th>OSCSELS</th>
<th>AMPHS1</th>
<th>AMPHS0</th>
<th>AMPH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>x</td>
<td>0/1</td>
<td>1</td>
<td>0/1</td>
<td>0/1</td>
<td>x</td>
</tr>
</tbody>
</table>

③ Clear (0) the XTSTOP bit of the CSC register to start oscillating the XT1 oscillator. After the external clock is input to the external clock signal, to clear (0) the XTSTOP bit.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>x</th>
<th>x</th>
<th>x</th>
<th>x</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

④ Use the timer function or another function to wait for oscillation of the subsystem clock oscillator to stabilize using software. This application note is counted using a 12-bit interval timer. External clock is not required oscillation stabilization wait.

⑤ Set(1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

<table>
<thead>
<tr>
<th>CKC</th>
<th>CLS</th>
<th>CSS</th>
<th>MCS</th>
<th>MCM0</th>
<th>0</th>
<th>0</th>
<th>MCS1</th>
<th>MCM1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

⑥ Confirm that the CLS bit of the CKC register has changed to 1, then set (1) HIOSTOP and stop the high-speed on-chip oscillator.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>x</th>
<th>x</th>
<th>x</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.5 Changing from middle-speed on-chip oscillator clock to high-speed on-chip oscillator clock

When changing the CPU clock from the middle-speed on-chip oscillator clock to the high-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the high-speed on-chip oscillator clock to $f_{cLK}$ using the system clock control register (CKC). Confirm that the status of the main on-chip oscillator clock status has switched to the high-speed on-chip oscillator clock, and then stop the middle-speed on-chip oscillator.

① Clear (0) the HIOSTOP bit of the CSC register, and then start oscillating the high-speed on-chip oscillator.

<table>
<thead>
<tr>
<th>Register Setting Values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

② Use a software wait to wait for the oscillation of the high-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 65μs) using the timer function or another function.

③ Clear(0) the MCM1 bit of the CKC register to specify the high-speed on-chip oscillator clock as the main on-chip oscillator clock.

<table>
<thead>
<tr>
<th>CKC</th>
<th>CLS</th>
<th>CSS</th>
<th>MCS</th>
<th>MCM0</th>
<th>0</th>
<th>0</th>
<th>MCS1</th>
<th>MCM1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

④ Confirm that the MCS1 bit of the CKC register has changed to 0, clear (0) the MIOEN bit and stop the oscillating the middle-speed on-chip oscillator.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
1.1.6 Changing from middle-speed on-chip oscillator clock to low-speed on-chip oscillator clock

When changing the CPU clock from the middle-speed on-chip oscillator clock to the low-speed on-chip oscillator clock, start the oscillation using the subsystem clock select register (CKSEL), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the low-speed on-chip oscillator clock to $f_{CLK}$ using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the subsystem clock, and then stop the middle-speed on-chip oscillator.

1. Set (1) the SELLOSC bit of the CKSEL register, and then start oscillating the low-speed on-chip oscillator.

```
<table>
<thead>
<tr>
<th>CKSEL</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
```

2. Use a software wait to wait for the oscillation of the low-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 210µs) using the timer function or another function. In this application note is always to operate the low-speed on-chip oscillator. For this reason, it does not perform oscillation stabilization wait of the low-speed on-chip oscillator.

3. Set (1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

```
<table>
<thead>
<tr>
<th>CKC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
```

4. Confirm that the CLS bit of the CKC register has changed to 1, clear (0) the MIOEN bit and stop the oscillating the middle-speed on-chip oscillator.

```
<table>
<thead>
<tr>
<th>CSC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.7 Changing from middle-speed on-chip oscillator clock to high-speed system clock

When changing the CPU clock from the middle-speed on-chip oscillator clock to the high-speed system clock, set the oscillator and start oscillation using the following registers: clock operation mode control register (CMC), oscillation stabilization time select register (OSTS), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the oscillation stabilization time counter status register (OSTC).

After the oscillation stabilizes, set the high-speed system clock to \( f_{CLK} \) using the system clock control register (CKC).

Confirm that the status of the main system clock has changed to the high-speed system clock, and then stop the middle-speed on-chip oscillator.

1. Set (1) the OSCSEL bit of the CMC register (when \( f_x > 10 \text{MHz} \), set (1) the AMPH bit) to operate the X1 oscillator. Set (1) the EXCLK bit and OSCSEL bit when using the external clock.

<table>
<thead>
<tr>
<th>CMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXCLK</td>
</tr>
<tr>
<td>0/1</td>
</tr>
</tbody>
</table>
   
   AMPH bit: clear to 0 when the X1 oscillation clock is 10 MHZ or lower.

2. Using the OSTS register, select the oscillation stabilization time of the X1 oscillation circuit. This setting does not have to exist at the time the external clock is used.

   Example: Set the following values for a wait of at least 102\( \mu \)s based on a 10 MHz resonator.

<table>
<thead>
<tr>
<th>OSTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td>0 0 0 0 0 OSTS2 OSTS1 OSTS0</td>
</tr>
<tr>
<td>0 0 0 0 0 0 1 0</td>
</tr>
</tbody>
</table>

3. Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator. After the external clock is input to the external clock signal, to clear (0) the MSTOP bit.

<table>
<thead>
<tr>
<th>CSC</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTOP</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

4. Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize. External clock is not required oscillation stabilization wait.

   Example: Wait until the bits reach the following values for a wait of at least 102\( \mu \)s based on a 10 MHz resonator.

<table>
<thead>
<tr>
<th>OSTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td>MOST8 MOST9 MOST10 MOST11 MOST13 MOST15 MOST17 MOST18</td>
</tr>
<tr>
<td>1 1 1 0 0 0 0 0</td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
⑤ Set(1) the MCM0 bit of the CKC register to specify the high-speed system clock as the main system clock.

<table>
<thead>
<tr>
<th>CKC</th>
<th>CLS</th>
<th>CSS</th>
<th>MCS</th>
<th>MCM0</th>
<th>0</th>
<th>0</th>
<th>MCS1</th>
<th>MCM1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

⑥ Confirm that the MCS bit of the CKC register has changed to 1, clear (0) the MIOEN bit and stop the oscillating the middle-speed on-chip oscillator.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIPOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit;
- -: reserved bits or unassigned bit
1.1.8 Changing from middle-speed on-chip oscillator clock to sub clock

When changing the CPU clock from the middle-speed on-chip oscillator clock to the sub clock, set the oscillator and start oscillation using the following registers: subsystem clock supply mode control register (OSMC), clock operation mode control register (CMC), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the timer or similar function. After the oscillation stabilizes, set the sub clock to $f_{CLK}$ using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock has switched to the subsystem clock, and then stop the middle-speed on-chip oscillator.

1. In this application, a 12-bit interval timer is used to count the oscillation stabilization time for the subsystem clock resonator. Set the WUTMMCK0 to 1 to use the low-speed on-chip oscillator clock as the count clock for the 12-bit interval timer. To run only the real-time clock and 12-bit interval timer on the subsystem clock in the STOP mode or HALT mode (during CPU operation with the subsystem clock), set the RTCLPC bit to 1.

<table>
<thead>
<tr>
<th>OSMC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTCLPC</td>
<td>0</td>
<td>0</td>
<td>WUTMMCK0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

2. Set(1) the OSCSELS bit of the CMC register to operate the XT1 oscillator. Set (1) the EXCLKS bit and OSCSELS bit when using the external clock.

<table>
<thead>
<tr>
<th>CMC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXCLK</td>
<td>OSCSEL</td>
<td>EXCLKS</td>
<td>OSCSELS</td>
<td>0</td>
<td>AMPHS1</td>
<td>AMPHS0</td>
<td>AMPH</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>0/1</td>
<td>1</td>
<td>0</td>
<td>0/1</td>
<td>0/1</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

3. Clear (0) the XTSTOP bit of the CSC register to start oscillating the XT1 oscillator. After the external clock is input to the external clock signal, to clear (0) the XTSTOP bit.

<table>
<thead>
<tr>
<th>CSC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

4. Use the timer function or another function to wait for oscillation of the subsystem clock oscillator to stabilize using software. This application note is counted using a 12-bit interval timer. External clock is not required oscillation stabilization wait.

5. Set(1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

<table>
<thead>
<tr>
<th>CKC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLS</td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</td>
<td>0</td>
<td>0</td>
<td>MCS1</td>
<td>MCM1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

6. Confirm that the CLS bit of the CKC register has changed to 1, then clear (0) MIOEN and stop the middle-speed on-chip oscillator.

<table>
<thead>
<tr>
<th>CSC</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit;
- - : reserved bits or unassigned bit
1.1.9 Changing from low-speed on-chip oscillator clock to high-speed on-chip oscillator clock

When changing the CPU clock from the low-speed on-chip oscillator clock to the high-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the high-speed on-chip oscillator clock to $f_{CLK}$ using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the main system clock, and then stop the low-speed on-chip oscillator.

1. Clear (0) the HIOSTOP bit of the CSC register, and then start oscillating the high-speed on-chip oscillator.

   CSC
<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>0</td>
</tr>
</tbody>
</table>

2. Use a software wait to wait for the oscillation of the high-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 65µs) using the timer function or another function.

3. Clear (0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

   CKC
<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLS</td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</td>
<td>0</td>
<td>0</td>
<td>MCS1</td>
<td>MCM1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

4. Confirm that the CLS bit of the CKC register has changed to 0, clear (0) the SELLOSC bit of the subsystem clock select register (CKSEL) and stop the oscillating the low-speed on-chip oscillator. In this application note, for WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) is 1, does not stop the low-speed on-chip oscillator.

   CKSEL
<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>SELLOSC</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock.

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.10 Changing from low-speed on-chip oscillator clock to middle-speed on-chip oscillator clock

When changing the CPU clock from the low-speed on-chip oscillator clock to the middle-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the middle-speed on-chip oscillator clock to fCLK using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the main system clock, and then stop the low-speed on-chip oscillator.

① Set (1) the MIOEN bit of the CSC register, and then start oscillating the middle-speed on-chip oscillator.

| CSC | MSTOP | XTSTOP | 0 | 0 | 0 | 0 | MIOEN | HIROS
|-----|-------|--------|---|---|---|---|-------|-----
|     | x     | x      | 0 | 0 | 0 | 0 | 1     | x   |

② Use a software wait to wait for the oscillation of the middle-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 4µs) using the timer function or another function.

③ Clear(0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

| CKC | CLS | CSS | MCS | MCM0 | 0 | 0 | MCS1 | MCM1
|-----|-----|-----|-----|------|---|---|------|-----
|     | 1   | 0   | 0   | 0    | 0 | 0 | 1    | 1   |

④ Confirm that the CLS bit of the CKC register has changed to 0, clear (0) the SELLOSC bit of the subsystem clock select register(CKSEL) and stop the oscillating the low-speed on-chip oscillator. In this application note, for WUTMMCK0 bit of the subsystem clock supply mode control register(OSMC) is 1, does not stop the low-speed on-chip oscillator.

| CKSEL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SELLOSC
|-------|---|---|---|---|---|---|---|------
|       | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0    |

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.11 Changing from low-speed on-chip oscillator clock to high-speed system clock

When changing the CPU clock from the low-speed on-chip oscillator clock to the high-speed system clock, set the oscillator and start oscillation using the following registers: clock operation mode control register (CMC), oscillation stabilization time select register (OSTS), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the oscillation stabilization time counter status register (OSTC).

After the oscillation stabilizes, set the high-speed system clock to $f_{CLK}$ using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock has changed to the main system clock, and then stop the low-speed on-chip oscillator.

1. Set (1) the OSCSEL bit of the CMC register (when $f_x > 10$MHz, set (1) the AMPH bit) to operate the X1 oscillator. Set (1) the EXCLK bit and OSCSEL bit when using the external clock.

<table>
<thead>
<tr>
<th>CMC</th>
<th>EXCLK</th>
<th>OSCSEL</th>
<th>EXCLKS</th>
<th>OSCSELS</th>
<th>AMPHS1</th>
<th>AMPHS0</th>
<th>AMPH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/1</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>0/1</td>
</tr>
</tbody>
</table>

   AMPH bit: clear to 0 when the X1 oscillation clock is 10 MHz or lower.

2. Using the OSTS register, select the oscillation stabilization time of the X1 oscillation circuit. This setting does not have to exist at the time the external clock is used.

   Example: Set the following values for a wait of at least 102 μs based on a 10 MHz resonator.

<table>
<thead>
<tr>
<th>OSTS</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

3. Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator. After the external clock is input to the external clock signal, to clear (0) the MSTOP bit.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

4. Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize. External clock is not required oscillation stabilization wait.

   Example: Wait until the bits reach the following values for a wait of at least 102 μs based on a 10 MHz resonator.

<table>
<thead>
<tr>
<th>OSTC</th>
<th>MOST8</th>
<th>MOST9</th>
<th>MOST10</th>
<th>MOST11</th>
<th>MOST13</th>
<th>MOST15</th>
<th>MOST17</th>
<th>MOST18</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
5. Clear(0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

```
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
+-----+-----+-----+-----+-----+-----+-----+-----+
| CLS | CSS | MCS | MCM0| 0   | 0   | MCS1| MCM1|
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1   | 0   | 1   | 1   | 0   | 0   | 0   | 0   |
```

6. Confirm that the CLS bit of the CKC register has changed to 0, clear (0) the SELLOSC bit of the subsystem clock select register (CKSEL) and stop the oscillating the low-speed on-chip oscillator. In this application note, for WUTMMCK0 bit of the subsystem clock supply mode control register (OSMC) is 1, does not stop the low-speed on-chip oscillator.

```
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7   | 6   | 5   | 4   | 3   | 2   | 1   | 0   |
+-----+-----+-----+-----+-----+-----+-----+-----+
| CKSEL|     |     |     |     |     |     | SELLOSC|
+-----+-----+-----+-----+-----+-----+-----+-----+
| 0   | 0   | 0   | 0   | 0   | 0   | 0   | 0   |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock.

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.12 Changing from high-speed system clock to high-speed on-chip oscillator clock

When changing the CPU clock from the high-speed system clock to the high-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the high-speed on-chip oscillator clock to \( f_{\text{CLK}} \) using the system clock control register (CKC). Confirm that the status of the main system clock status has switched to the main on-chip oscillator clock, and then stop the X1 oscillator.

① Clear (0) the HIOSTOP bit of the CSC register, and then start oscillating the high-speed on-chip oscillator.

```
<table>
<thead>
<tr>
<th></th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSC</td>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>0</td>
</tr>
</tbody>
</table>
```

② Use a software wait to wait for the oscillation of the high-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 65µs) using the timer function or another function.

③ Clear(0) the MCM0 bit of the CKC register to specify the main on-chip oscillator clock as the main system clock.

```
<table>
<thead>
<tr>
<th></th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKC</td>
<td>CLS</td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</td>
<td>0</td>
<td>0</td>
<td>MCS1</td>
<td>MCM1</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

④ Confirm that the MCS bit of the CKC register has changed to 0, set (1) the MSTOP bit and stop the oscillating the X1 oscillator.

```
<table>
<thead>
<tr>
<th></th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSC</td>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.13 Changing from high-speed system clock to middle-speed on-chip oscillator clock

When changing the CPU clock from the high-speed system clock to the middle-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the middle-speed on-chip oscillator clock to $f_{CLK}$ using the system clock control register (CKC). Confirm that the status of the main system clock status has switched to the main on-chip oscillator clock, and then stop the X1 oscillator.

① Set (1) the MIOEN bit of the CSC register, and then start oscillating the middle-speed on-chip oscillator.

```
<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
```

② Use a software wait to wait for the oscillation of the middle-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 4µs) using the timer function or another function.

③ Clear(0) the MCM0 bit of the CKC register to specify the main on-chip oscillator clock as the main system clock.

```
<table>
<thead>
<tr>
<th>CKC</th>
<th>CLS</th>
<th>CSS</th>
<th>MCS</th>
<th>MCM0</th>
<th>0</th>
<th>0</th>
<th>MCS1</th>
<th>MCM1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
```

④ Confirm that the MCS bit of the CKC register has changed to 0, set (1) the MSTOP bit and stop the oscillating the X1 oscillator.

```
<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
```

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.14 Changing from high-speed system clock to low-speed on-chip oscillator clock

When changing the CPU clock from the high-speed system clock to the low-speed on-chip oscillator clock, start the oscillation using the subsystem clock select register (CKSEL), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the low-speed on-chip oscillator clock to \( f_{\text{CLK}} \) using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the subsystem clock, and then stop the X1 oscillator.

1. Set (1) the SELLOSC bit of the CKSEL register, and then start oscillating the low-speed on-chip oscillator.

   \[
   \begin{array}{cccccccc}
   7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
   \hline
   0 & 0 & 0 & 0 & 0 & 0 & 0 & \text{SELOSC} \\
   0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
   \end{array}
   \]

2. Use a software wait to wait for the oscillation of the low-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 210\( \mu \text{s} \)) using the timer function or another function. In this application note is always to operate the low-speed on-chip oscillator. For this reason, it does not perform oscillation stabilization wait of the low-speed on-chip oscillator.

3. Set(1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

   \[
   \begin{array}{cccccccc}
   7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
   \hline
   \text{CLS} & \text{CSS} & \text{MCS} & \text{MCM0} & 0 & 0 & \text{MCS1} & \text{MCM1} \\
   0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
   \end{array}
   \]

4. Confirm that the CLS bit of the CKC register has changed to 1, set (1) the MSTOP bit and stop the oscillating the X1 oscillator.

   \[
   \begin{array}{cccccccc}
   7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\
   \hline
   \text{MSTOP} & \text{XTSTOP} & 0 & 0 & 0 & 0 & \text{MIOEN} & \text{HIOSTOP} \\
   1 & x & 0 & 0 & 0 & 0 & x & x \\
   \end{array}
   \]

Register setting values:
- x: unused bit; blank space; unchanged bit;
- -: reserved bits or unassigned bit
1.1.15 Changing from high-speed system clock to sub clock

When changing the CPU clock from the high-speed system clock to the sub clock, set the oscillator and start oscillating using the following registers: subsystem clock supply mode control register (OSMC), clock operation mode control register (CMC), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the timer or similar function. After the oscillation stabilizes, set the sub clock to \( f_{CLK} \) using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock has switched to the subsystem clock, and then stop the X1 oscillator.

1. In this application, a 12-bit interval timer is used to count the oscillation stabilization time for the subsystem clock resonator. Set the WUTMMCK0 to 1 to use the low-speed on-chip oscillator clock as the count clock for the 12-bit interval timer. To run only the real-time clock and 12-bit interval timer on the subsystem clock in the STOP mode or HALT mode (during CPU operation with the subsystem clock), set the RTCLPC bit to 1.

   ![OSMC Register](7 6 5 4 3 2 1 0 RTCLPC 0 0 WUTMMCK0 0 0 0 0 x x x x)

2. Clear (0) the XTSTOP bit of the CSC register to start oscillating the XT1 oscillator. After the external clock is input to the external clock signal, to clear (0) the XTSTOP bit.

   ![CSC Register](7 6 5 4 3 2 1 0 MSTOP XTSTOP 0 0 0 0 0 MIOEN HIOSTOP 0 0 0 1 0 0 x x)

3. Use the timer function or another function to wait for oscillation of the subsystem clock oscillator to stabilize using software. This application note is counted using a 12-bit interval timer. External clock is not required oscillation stabilization wait.

4. Set(1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

   ![CKC Register](7 6 5 4 3 2 1 0 CLS CSS MCS MCM0 0 0 MCS1 MCM1 0 1 1 1 0 0 0 0)

5. Confirm that the CLS bit of the CKC register has changed to 1, then set (1) MSTOP and stop the X1 oscillator.

   ![CSC Register](7 6 5 4 3 2 1 0 MSTOP XTSTOP 0 0 0 0 0 MIOEN HIOSTOP 1 0 0 0 0 1 x x)

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
### Changing from sub clock to high-speed on-chip oscillator clock

When changing the CPU clock from the sub clock to the high-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the high-speed on-chip oscillator clock to $f_{CLK}$ using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the main system clock, and then stop the XT1 oscillator.

1. Clear (0) the HIOSTOP bit of the CSC register, and then start oscillating the high-speed on-chip oscillator.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>0</td>
</tr>
</tbody>
</table>

2. Use a software wait to wait for the oscillation of the high-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 65µs) using the timer function or another function.

3. Clear(0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

<table>
<thead>
<tr>
<th>CKC</th>
<th>CLS</th>
<th>CSS</th>
<th>MCS</th>
<th>MCM0</th>
<th>0</th>
<th>0</th>
<th>MCS1</th>
<th>MCM1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

4. Confirm that the CLS bit of the CKC register has changed to 0, set (1) the XTSTOP bit and stop the oscillating the XT1 oscillator.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>0</td>
</tr>
</tbody>
</table>

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock.

---

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
### 1.1.17 Changing from sub clock to middle-speed on-chip oscillator clock

When changing the CPU clock from the sub clock to the middle-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the middle-speed on-chip oscillator clock to \( f_{\text{CLK}} \) using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the main system clock, and then stop the XT1 oscillator.

1. Set (1) the MIOEN bit of the CSC register, and then start oscillating the middle-speed on-chip oscillator.

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>x</td>
</tr>
</tbody>
</table>

2. Use a software wait to wait for the oscillation of the middle-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 4µs) using the timer function or another function.

3. Clear(0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLS</td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</td>
<td>0</td>
<td>0</td>
<td>MCS1</td>
<td>MCM1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

4. Confirm that the CLS bit of the CKC register has changed to 0, set (1) the XTSTOP bit and stop the oscillating XT1 oscillator.

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>x</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>x</td>
</tr>
</tbody>
</table>

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock.

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.18 Changing from sub clock to high-speed system clock

When changing the CPU clock from the sub clock to the high-speed system clock, set the oscillator and start oscillation using the following registers: clock operation mode control register (CMC), oscillation stabilization time select register (OSTS), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the oscillation stabilization time counter status register (OSTC).

After the oscillation stabilizes, set the high-speed system clock to fCLK using the system clock control register (CKC).

Confirm that the status of the CPU/peripheral hardware clock has changed to the main system clock, and then stop the XT1 oscillator.

① Using the OSTS register, select the oscillation stabilization time of the X1 oscillation circuit. This setting does not have to exist at the time the external clock is used.

Example: Set the following values for a wait of at least 102μs based on a 10 MHz resonator.

<table>
<thead>
<tr>
<th>OSTS</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>OSTS2</td>
<td>OSTS1</td>
<td>OSTS0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

② Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator. After the external clock is input to the external clock signal, to clear (0) the MSTOP bit.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

③ Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize. External clock is not required oscillation stabilization wait.

Example: Wait until the bits reach the following values for a wait of at least 102μs based on a 10 MHz resonator.

<table>
<thead>
<tr>
<th>OSTC</th>
<th>MOST8</th>
<th>MOST9</th>
<th>MOST10</th>
<th>MOST11</th>
<th>MOST13</th>
<th>MOST15</th>
<th>MOST17</th>
<th>MOST18</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

④ Clear(0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

<table>
<thead>
<tr>
<th>CKC</th>
<th>CLS</th>
<th>CSS</th>
<th>MCS</th>
<th>MCM0</th>
<th>0</th>
<th>0</th>
<th>MCS1</th>
<th>MCM1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

⑤ Confirm that the CLS bit of the CKC register has changed to 0, then set (1) XTSTOP and stop the XT1 oscillator.

<table>
<thead>
<tr>
<th>CSC</th>
<th>MSTOP</th>
<th>XTSTOP</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock

Register setting values:

x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
2. Operation Confirmation Conditions

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

### Table 2.1 Operation Confirmation Conditions

<table>
<thead>
<tr>
<th>Item</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCU used</td>
<td>RL78/I1D (R5F117GC)</td>
</tr>
</tbody>
</table>
| Operating frequencies    | • High-speed on-chip oscillator clock: 24MHz  
• Middle-speed on-chip oscillator clock: 4MHz  
• Low-speed on-chip oscillator clock: 15kHz  
• High-speed system clock: 20MHz  
• Sub clock: 32.768kHz  
• CPU/peripheral hardware clock: 24MHz/20MHz/4MHz/32.768kHz/15kHz\(^\text{Note}\) |
| Operating voltage        | 3.0V (operating range 2.9V to 3.6V)  
LVD operations (\(V_{LVD}\)): reset mode 2.81V (2.76V to 2.87V)                                                                 |
| Integrated development   | IAR Embedded Workbench V2.21.5                                                                                                             |
| environment              |                                                                                                                                            |
| C compiler               | IAR C/C++ Compiler V2.21.1.18333                                                                                                           |
| Board used               | Renesas Electronics Corp.  
RL78/I1D Target board (RTE5117GC0TGB0000R)                                                                                             |

Note: CPU/peripheral hardware clock settings are changed in the application.

3. Related Application Notes

Application notes related to this document are shown below. Please refer to these as needed.

RL78/I1D Operation State Switching IAR (R01AN3597E) Application Note
4. Hardware Explanation

4.1 Hardware Configuration Example

Figure 4.1 shows an example of the hardware configuration used in this application note.

![Figure 4.1 Hardware Configuration](image)

Note: 1. This simplified circuit diagram was created to show an overview of connections only.
   When actually designing your circuit, make sure the design includes sufficient pin processing and meets electrical characteristic requirements.
   (Connect each input-only port to VDD or VSS through a resistor.)
2. If a pin name starts with EVSS, connect the pin to VSS, if it starts with EVDD, connect it to VDD.
3. Make VDD higher than the RESET release voltage (V_LVD) set in LVD.
4. P60 and P61 are N-Ch open drain output ports. Because a through current may flow in Hi-Z, please be connected to VDD through a resistor.

4.2 Used Pin List

Table 4.1 provides List of Pins and Functions

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Input/Output</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>P137/INTP0</td>
<td>Input</td>
<td>Switch (SW1) input port</td>
</tr>
<tr>
<td>P20/AIN13</td>
<td>Input</td>
<td>A/D converter analog input port</td>
</tr>
<tr>
<td>P60</td>
<td>Output</td>
<td>LED (LED1) control port</td>
</tr>
<tr>
<td>P61</td>
<td>Output</td>
<td>LED (LED2) control port</td>
</tr>
<tr>
<td>P50</td>
<td>Output</td>
<td>LED (LED3) control port</td>
</tr>
<tr>
<td>P51</td>
<td>Output</td>
<td>LED (LED4) control port</td>
</tr>
<tr>
<td>P52</td>
<td>Output</td>
<td>LED (LED5) control port</td>
</tr>
</tbody>
</table>
5. Software Explanation

5.1 Operation Outline

This application enables the user to change the CPU clock and the operation mode using switch input. The CPU clock and the operating mode is changed in the order of 1 to 39 of Figure 1.1 operation mode status transition diagram.

(1) Input/output port initialization

- P50-P52 and P60-P61 pins: set as output ports (use to control LEDs)
- P137/INTP0 pin: set as input port (use for switch input)
- P20/ANI13 pin: set as analog input port (use as A/D conversion analog input channel)

(2) Clock generator initialization

<Setting conditions>

- Set the flash operation mode to HS (high-speed main) mode using user option byte (00C2H/010C2H).)
- High-speed on-chip oscillator clock frequency: set to 24 MHz
- Set the operation mode of the subsystem clock pin to XT1 oscillation, and connect a crystal resonator to the XT1/123 and XT2/EXCLKS/P124 pins.
- Set the oscillation mode of the XT1 oscillator to ultra-low power consumption oscillation. (Select the optimal oscillation mode for the oscillator connected to the board.)
- Set the operation mode of the high-speed system clock pin to X1 oscillation, and connect a crystal resonator to the X1/P121 and X2/EXCLK/P122 pins.
- Select the main system clock (f_MAIN) as the CPU/peripheral hardware clock (f_CLK).

(3) Interrupt processing initialization

- Set the INTP0 pin valid edge to falling edge and enable switch input.
- Use the 12-bit interval timer to confirm switch input. The voltage level of the pin is checked approximately every 5 ms. If the voltage level matches twice consecutively, the switch input is recognized as valid (prevents chattering).
The CPU clock and operation mode change as follows each time the falling edge of a signal (switch) input to the P137/INTP0 pin is detected. The following is the CPU clock and operation mode after the switch is pressed.

### Table 5.1 LED status (after the switch is pressed) (1/2)

<table>
<thead>
<tr>
<th>CPU clock</th>
<th>Operation mode</th>
<th>LED1</th>
<th>LED2</th>
<th>LED3</th>
<th>LED4</th>
<th>LED5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) High-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(2) Sub clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(3) Sub clock</td>
<td>HALT mode</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(4) Sub clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(5) High-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(6) High-speed system clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(7) Sub clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(8) High-speed system clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(9) High-speed system clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(10) High-speed system clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(11) High-speed system clock</td>
<td>Normal operation mode</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(12) High-speed system clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(13) High-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(14) High-speed on-chip oscillator clock</td>
<td>HALT mode</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(15) High-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(16) High-speed on-chip oscillator clock</td>
<td>STOP mode</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(17) High-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(18) High-speed on-chip oscillator clock</td>
<td>SNOOZE mode</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(19) High-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(20) Middle-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(21) High-speed system clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(22) Low-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>(23) High-speed system clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(24) Middle-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
</tbody>
</table>
## Table 5.2  LED Status (after the switch is pressed) (2/2)

<table>
<thead>
<tr>
<th>CPU clock</th>
<th>Operation mode</th>
<th>LED1</th>
<th>LED2</th>
<th>LED3</th>
<th>LED4</th>
<th>LED5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(25) Middle-speed on-chip oscillator clock</td>
<td>HALT mode</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(26) Middle-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(27) Middle-speed on-chip oscillator clock</td>
<td>STOP mode</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(28) Middle-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(29) Middle-speed on-chip oscillator clock</td>
<td>SNOOZE mode</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(30) Middle-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(31) Sub clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(32) Middle-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(33) Low-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>(34) Middle-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>(35) High-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(36) Low-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>(37) Low-speed on-chip oscillator clock</td>
<td>HALT mode</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>(38) Low-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>(39) High-speed on-chip oscillator clock</td>
<td>Normal operation mode</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
</tbody>
</table>

After the CPU clock and operation mode have been changed according to steps 1 to 39 above, the falling edge of a signal (switch) input to the P137/INTP0 pin is detected, all LEDs are turned OFF, and the CPU goes to HALT mode (only RESET input in standby recovery).

In addition, if the CPU clock can’t be status transition to a certain period of time such as by oscillation failure of the crystal oscillator is all LEDs are turned OFF and end the status transition in error processing.

Note: Refer to the RL78/I1D User’s Manual for usage notes concerning this device.
5.2 Option Byte Settings

Table 5.3 lists the option byte settings.

<table>
<thead>
<tr>
<th>Address</th>
<th>Setting Value</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>000C0H/010C0H</td>
<td>01101110B</td>
<td>Watchdog timer operation is stopped (count is stopped after reset)</td>
</tr>
<tr>
<td>000C1H/010C1H</td>
<td>01111111B</td>
<td>LVD operation (V_LVD): reset mode Detection voltage: Rising edge 2.81 V/falling edge 2.75 V</td>
</tr>
<tr>
<td>000C2H/010C2H</td>
<td>11100000B</td>
<td>HS mode, High-speed on-chip oscillator clock: 24 MHz</td>
</tr>
<tr>
<td>000C3H/010C3H</td>
<td>10000100B</td>
<td>On-chip debugging enabled</td>
</tr>
</tbody>
</table>

5.3 Variables

Table 5.4 lists the global variables.

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable Name</th>
<th>Contents</th>
<th>Function Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-bit</td>
<td>g_int_cnt</td>
<td>Number of interval signal detection interrupts for 12-bit interval timer</td>
<td>R_MAIN_BtoE, R_MAIN_EtoB, R_MAIN_BtoD, R_MAIN_DtoE, R_MAIN_EtoD, R_MAIN_DtoB, R_MAIN_BtoC, R_MAIN_CtoD, R_MAIN_DtoF, R_MAIN_FtoD, R_MAIN_DtoC, R_MAIN_CtoE, R_MAIN_EtoC, R_MAIN_CtoB, R_MAIN_BtoF, R_MAIN_CtoF, R_MAIN_FtoC, r_it_interrupt</td>
</tr>
<tr>
<td>8-bit</td>
<td>g_int_flg</td>
<td>Confirm the external interrupt generation detection flag</td>
<td>R_MAIN_NOP_Loop, r_intc0_interrupt</td>
</tr>
</tbody>
</table>
5.4 Functions (subroutines)

Table 5.5 and Table 5.6 lists the functions (subroutines).

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Outline</th>
<th>Number of operating mode status transition diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_MAIN_AtoB</td>
<td>Status transition processing from (A) to (B)</td>
<td>(1)</td>
</tr>
<tr>
<td>R_MAIN_BtoE</td>
<td>Status transition processing from (B) to (E)</td>
<td>(2)</td>
</tr>
<tr>
<td>R_MAIN_EtoO</td>
<td>Status transition processing from (E) to (O)</td>
<td>(3)</td>
</tr>
<tr>
<td>R_MAIN_OtoE</td>
<td>Status transition processing from (O) to (E)</td>
<td>(4)</td>
</tr>
<tr>
<td>R_MAIN_EtoB</td>
<td>Status transition processing from (E) to (B)</td>
<td>(5)</td>
</tr>
<tr>
<td>R_MAIN_BtoD</td>
<td>Status transition processing from (B) to (D)</td>
<td>(6)</td>
</tr>
<tr>
<td>R_MAIN_DtoE</td>
<td>Status transition processing from (D) to (E)</td>
<td>(7)</td>
</tr>
<tr>
<td>R_MAIN_EtoD</td>
<td>Status transition processing from (E) to (D)</td>
<td>(8)</td>
</tr>
<tr>
<td>R_MAIN_DtoM</td>
<td>Status transition processing from (D) to (M)</td>
<td>(9)</td>
</tr>
<tr>
<td>R_MAIN_MtoD</td>
<td>Status transition processing from (M) to (D)</td>
<td>(10)</td>
</tr>
<tr>
<td>R_MAIN_DtoN</td>
<td>Status transition processing from (D) to (N)</td>
<td>(11)</td>
</tr>
<tr>
<td>R_MAIN_NtoD</td>
<td>Status transition processing from (N) to (D)</td>
<td>(12)</td>
</tr>
<tr>
<td>R_MAIN_DtoB</td>
<td>Status transition processing from (D) to (B)</td>
<td>(13)</td>
</tr>
<tr>
<td>R_MAIN_BtoI</td>
<td>Status transition processing from (B) to (I)</td>
<td>(14)</td>
</tr>
<tr>
<td>R_MAIN_ItoB</td>
<td>Status transition processing from (I) to (B)</td>
<td>(15)</td>
</tr>
<tr>
<td>R_MAIN_GtoB</td>
<td>Status transition processing from (G) to (B)</td>
<td>(16)</td>
</tr>
<tr>
<td>R_MAIN_BtoH</td>
<td>Status transition processing from (B) to (H)</td>
<td>(17)</td>
</tr>
<tr>
<td>R_MAIN_HtoB</td>
<td>Status transition processing from (H) to (B)</td>
<td>(18)</td>
</tr>
<tr>
<td>R_MAIN_BtoC</td>
<td>Status transition processing from (B) to (C)</td>
<td>(19)</td>
</tr>
<tr>
<td>R_MAIN_CtoD</td>
<td>Status transition processing from (C) to (D)</td>
<td>(20)</td>
</tr>
<tr>
<td>R_MAIN_DtoF</td>
<td>Status transition processing from (D) to (F)</td>
<td>(21)</td>
</tr>
<tr>
<td>R_MAIN_FtoD</td>
<td>Status transition processing from (F) to (D)</td>
<td>(22)</td>
</tr>
<tr>
<td>R_MAIN_DtoC</td>
<td>Status transition processing from (D) to (C)</td>
<td>(23)</td>
</tr>
<tr>
<td>R_MAIN_CtoJ</td>
<td>Status transition processing from (C) to (J)</td>
<td>(24)</td>
</tr>
<tr>
<td>R_MAIN_JtoC</td>
<td>Status transition processing from (J) to (C)</td>
<td>(25)</td>
</tr>
<tr>
<td>R_MAIN_CtoK</td>
<td>Status transition processing from (C) to (K)</td>
<td>(26)</td>
</tr>
<tr>
<td>R_MAIN_KtoC</td>
<td>Status transition processing from (K) to (C)</td>
<td>(27)</td>
</tr>
<tr>
<td>R_MAIN_CtoL</td>
<td>Status transition processing from (C) to (L)</td>
<td>(28)</td>
</tr>
<tr>
<td>R_MAIN_LtoC</td>
<td>Status transition processing from (L) to (C)</td>
<td>(29)</td>
</tr>
<tr>
<td>R_MAIN_CtoE</td>
<td>Status transition processing from (C) to (E)</td>
<td>(30)</td>
</tr>
<tr>
<td>R_MAIN_EtoC</td>
<td>Status transition processing from (E) to (C)</td>
<td>(31)</td>
</tr>
<tr>
<td>R_MAIN_CtoF</td>
<td>Status transition processing from (C) to (F)</td>
<td>(32)</td>
</tr>
<tr>
<td>R_MAIN_FtoC</td>
<td>Status transition processing from (F) to (C)</td>
<td>(33)</td>
</tr>
<tr>
<td>R_MAIN_CtoB</td>
<td>Status transition processing from (C) to (B)</td>
<td>(34)</td>
</tr>
<tr>
<td>R_MAIN_BtoF</td>
<td>Status transition processing from (B) to (F)</td>
<td>(35)</td>
</tr>
<tr>
<td>R_MAIN_FtoP</td>
<td>Status transition processing from (F) to (P)</td>
<td>(36)</td>
</tr>
<tr>
<td>R_MAIN_PtoF</td>
<td>Status transition processing from (P) to (F)</td>
<td>(37)</td>
</tr>
<tr>
<td>R_MAIN_FtoB</td>
<td>Status transition processing from (F) to (B)</td>
<td>(38)</td>
</tr>
<tr>
<td>Function Name</td>
<td>Outline</td>
<td>Number of operating mode status transition diagram</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>R_MAIN_NOP_Loop</td>
<td>Continuous NOP instruction execution processing</td>
<td>—</td>
</tr>
<tr>
<td>R_MAIN_END</td>
<td>A/D converter setting</td>
<td>—</td>
</tr>
<tr>
<td>R_MAIN_ERROR</td>
<td>End processing of status transition</td>
<td>—</td>
</tr>
<tr>
<td>R_MAIN_Set_SnoozeOn</td>
<td>Error processing of status transition</td>
<td>—</td>
</tr>
<tr>
<td>r_intc0_interrupt</td>
<td>Confirm the external interrupt generation detection flag update processing</td>
<td>—</td>
</tr>
<tr>
<td>r_it_interrupt</td>
<td>12-bit interval timer interval signal detection interrupt count processing</td>
<td>—</td>
</tr>
<tr>
<td>r_adc_interrupt</td>
<td>SNOOZE mode release processing</td>
<td>—</td>
</tr>
</tbody>
</table>
### 5.5 Function (subroutine) Specifications

The following are the sample code functions (subroutines) used in this application note.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Outline</th>
<th>Declaration</th>
<th>Description</th>
<th>Argument</th>
<th>Return Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_MAIN_AtoB</td>
<td>Status transition processing from (A) to (B)</td>
<td>void R_MAIN_AtoB(void)</td>
<td>Control LED lighting. (CPU clock: high-speed on-chip oscillator clock)</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>R_MAIN_BtoE</td>
<td>Status transition processing from (B) to (E)</td>
<td>void R_MAIN_BtoE(void)</td>
<td>Change the CPU clock from high-speed on-chip oscillator clock to sub clock. After the clock is switched, control LED lighting.</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>R_MAIN_EtoO</td>
<td>Status transition processing from (E) to (O)</td>
<td>void R_MAIN_EtoO(void)</td>
<td>Control LED lighting, transition to HALT mode. (CPU clock stopped (when using sub clock))</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>R_MAIN_OtoE</td>
<td>Status transition processing from (O) to (E)</td>
<td>void R_MAIN_OtoE(void)</td>
<td>Control LED lighting (CPU clock: sub clock)</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>R_MAIN_EtoB</td>
<td>Status transition processing from (E) to (B)</td>
<td>void R_MAIN_EtoB(void)</td>
<td>Change the CPU clock from sub clock to high-speed on-chip oscillator clock. After the clock is switched, control LED lighting.</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>
### R_MAIN_BtoD

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (B) to (D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_BtoD(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Change the CPU clock from high-speed on-chip oscillator clock to high-speed system clock. After the clock is switched, control LED lighting.</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>

### R_MAIN_DtoE

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (D) to (E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_DtoE(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Change the CPU clock from high-speed system clock to sub clock. After the clock is switched, control LED lighting.</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>

### R_MAIN_EtoD

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (E) to (D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_EtoD(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Change the CPU clock from sub clock to high-speed system clock. After the clock is switched, control LED lighting.</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>

### R_MAIN_DtoM

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (D) to (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_DtoM(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting, then transition to HALT mode. (CPU clock stopped (when using high-speed system clock))</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>

### R_MAIN_MtoD

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (M) to (D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_MtoD(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting. (CPU clock: high-speed system clock)</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>
### [Function Name] R_MAIN_DtoN

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (D) to (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_DtoN(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting, then transition to STOP mode. (Stop CPU clock when using high-speed system clock)</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>

### [Function Name] R_MAIN_NtoD

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (N) to (D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_NtoD(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting. (CPU clock: high-speed system clock)</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>

### [Function Name] R_MAIN_DtoB

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (D) to (B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_DtoB(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Change the CPU clock from high-speed system clock to high-speed on-chip oscillator clock. After the clock is switched, control LED lighting.</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>

### [Function Name] R_MAIN_BtoI

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (B) to (I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_BtoI(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting, then transition to HALT mode. (CPU clock stopped when using high-speed on-chip oscillator clock)</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>

### [Function Name] R_MAIN_ItoB

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (I) to (B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_ItoB(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting. (CPU clock: high-speed on-chip oscillator clock)</td>
</tr>
<tr>
<td>Argument</td>
<td>None</td>
</tr>
<tr>
<td>Return Value</td>
<td>None</td>
</tr>
<tr>
<td>Notes</td>
<td>None</td>
</tr>
</tbody>
</table>
[Function Name] R_MAIN_BtoG

Outline  | Status transition processing from (B) to (G)
Declaration | void R_MAIN_BtoG(void)
Description | Control LED lighting, then transition to STOP mode.
| (CPU clock stopped (when using high-speed on-chip oscillator clock))
Argument | None
Return Value | None
Notes | None

[Function Name] R_MAIN_GtoB

Outline  | Status transition processing from (G) to (B)
Declaration | void R_MAIN_GtoB(void)
Description | Control LED lighting.
| (CPU clock: high-speed on-chip oscillator clock)
Argument | None
Return Value | None
Notes | None

[Function Name] R_MAIN_BtoH

Outline  | Status transition processing from (B) to (H)
Declaration | void R_MAIN_BtoH(void)
Description | Set A/D converter and control LED lighting.
| Then, transition to SNOOZE mode.
Argument | None
Return Value | None
Notes | None

[Function Name] R_MAIN_HtoB

Outline  | Status transition processing from (H) to (B)
Declaration | void R_MAIN_HtoB(void)
Description | Set SNOOZE release and stop A/D converter.
| Then control LED lighting.
Argument | None
Return Value | None
Notes | None

[Function Name] R_MAIN_BtoC

Outline  | Status transition processing from (B) to (C)
Declaration | void R_MAIN_BtoC(void)
Description | Change the CPU clock from high-speed on-chip oscillator clock to middle-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument | None
Return Value | None
Notes | None
[Function Name] R_MAIN_CtoD

Outline  Status transition processing from (C) to (D)
Declaration  void R_MAIN_CtoD(void)
Description  Change the CPU clock from middle-speed on-chip oscillator clock to high-speed system clock. After the clock is switched, control LED lighting.
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_DtoF

Outline  Status transition processing from (D) to (F)
Declaration  void R_MAIN_DtoF(void)
Description  Change the CPU clock from high-speed system clock to low-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_FtoD

Outline  Status transition processing from (F) to (D)
Declaration  void R_MAIN_FtoD(void)
Description  Change the CPU clock from low-speed on-chip oscillator clock to high-speed system clock. After the clock is switched, control LED lighting.
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_DtoC

Outline  Status transition processing from (D) to (C)
Declaration  void R_MAIN_DtoC(void)
Description  Change the CPU clock from high-speed system clock to middle-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_CtoJ

Outline  Status transition processing from (C) to (J)
Declaration  void R_MAIN_CtoJ(void)
Description  Control LED lighting, then transition to HALT mode.
 Argument  None
Return Value  None
Notes  None
[Function Name] R_MAIN_JtoC

Outline  Status transition processing from (J) to (C)
Declaration void R_MAIN_JtoC(void)
Description Control LED lighting.
            (CPU clock: middle-speed on-chip oscillator clock)
Argument  None
Return Value None
Notes      None

[Function Name] R_MAIN_CtoK

Outline  Status transition processing from (C) to (K)
Declaration void R_MAIN_CtoK(void)
Description Control LED lighting, then transition to STOP mode.
            (CPU clock stopped (when using middle-speed on-chip oscillator clock))
Argument  None
Return Value None
Notes      None

[Function Name] R_MAIN_KtoC

Outline  Status transition processing from (K) to (C)
Declaration void R_MAIN_KtoC(void)
Description Control LED lighting.
            (CPU clock: middle-speed on-chip oscillator clock)
Argument  None
Return Value None
Notes      None

[Function Name] R_MAIN_CtoL

Outline  Status transition processing from (C) to (L)
Declaration void R_MAIN_CtoL(void)
Description Set A/D converter and control LED lighting.
            Then, transition to SNOOZE mode.
Argument  None
Return Value None
Notes      None

[Function Name] R_MAIN_LtoC

Outline  Status transition processing from (L) to (C)
Declaration void R_MAIN_LtoC(void)
Description Set SNOOZE release and stop A/D converter.
            Then control LED lighting.
Argument  None
Return Value None
Notes      None
[Function Name] R_MAIN_CtoE

Outline    Status transition processing from (C) to (E)
Declaration void R_MAIN_CtoE(void)
Description Change the CPU clock from middle-speed on-chip oscillator clock to sub clock. After
the clock is switched, control LED lighting.
Argument    None
Return Value None
Notes       None

[Function Name] R_MAIN_EtoC

Outline    Status transition processing from (E) to (C)
Declaration void R_MAIN_EtoC(void)
Description Change the CPU clock from sub clock to middle-speed on-chip oscillator clock. After
the clock is switched, control LED lighting.
Argument    None
Return Value None
Notes       None

[Function Name] R_MAIN_CtoF

Outline    Status transition processing from (C) to (F)
Declaration void R_MAIN_CtoF(void)
Description Change the CPU clock from middle-speed on-chip oscillator clock to low-speed
on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument    None
Return Value None
Notes       None

[Function Name] R_MAIN_FtoC

Outline    Status transition processing from (F) to (C)
Declaration void R_MAIN_FtoC(void)
Description Change the CPU clock from low-speed on-chip oscillator clock to middle-speed
on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument    None
Return Value None
Notes       None

[Function Name] R_MAIN_CtoB

Outline    Status transition processing from (C) to (B)
Declaration void R_MAIN_CtoB(void)
Description Change the CPU clock from middle-speed on-chip oscillator clock to high-speed
on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument    None
Return Value None
Notes       None
[Function Name] R_MAIN_BtoF

Outline  Status transition processing from (B) to (F)
Declaration  void R_MAIN_BtoF(void)
Description  Change the CPU clock from high-speed on-chip oscillator clock to low-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_FtoP

Outline  Status transition processing from (F) to (P)
Declaration  void R_MAIN_FtoP(void)
Description  Control LED lighting, then transition to HALT mode.  
               (CPU clock stopped (when using low-speed on-chip oscillator clock))
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_PtoF

Outline  Status transition processing from (P) to (F)
Declaration  void R_MAIN_PtoF(void)
Description  Control LED lighting.  
               (CPU clock: low-speed on-chip oscillator clock)
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_FtoB

Outline  Status transition processing from (F) to (B)
Declaration  void R_MAIN_FtoB(void)
Description  Change the CPU clock from low-speed on-chip oscillator clock to high-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_NOP_Loop

Outline  Continuous NOP instruction execution processing
Declaration  void R_MAIN_NOP_Loop(void)
Description  Execute NOP instruction continuously. End processing when external interrupt generation detection flag is confirmed.
Argument  None
Return Value  None
Notes  None
[Function Name] R_MAIN_END

Outline  End processing of status transition
Declaration  void R_MAIN_END(void)
Description  Disable interrupts. Control LED lighting (all off).
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_ERROR

Outline  Error processing of status transition
Declaration  void R_MAIN_ERROR(void)
Description  Disable interrupts. Control LED lighting (all off). Loop processing within the function(Return is only reset input).
Argument  None
Return Value  None
Notes  None

[Function Name] R_MAIN_AD_SnoozeOn

Outline  A/D converter setting
Declaration  void R_MAIN_AD_SnoozeOn(void)
Description  Set A/D converter to hardware trigger wait mode with 12-bit interval timer interrupt signal. Enable SNOOZE mode and transition to A/D conversion wait status.
Argument  None
Return Value  None
Notes  None

[Function Name] r_intc0_interrupt

Outline  External interrupt generation detection flag confirmation processing
Declaration  __interrupt void r_intc0_interrupt(void)
Description  Confirm external interrupt generation detection flag with generation of external interrupt. End processing when switch input level changes to high.
Argument  None
Return Value  None
Notes  None

[Function Name] r_it_interrupt

Outline  12-bit interval timer interval signal detection interrupt count processing
Declaration  __interrupt void r_it_interrupt(void)
Description  Increment the g_int_cnt each time the 12-bit interval timer interrupt signal detection interrupt is generated.
Argument  None
Return Value  None
Notes  None
<table>
<thead>
<tr>
<th><strong>Function Name</strong></th>
<th>r_adc_interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Outline</strong></td>
<td>SNOOZE mode release processing</td>
</tr>
<tr>
<td><strong>Declaration</strong></td>
<td>__interrupt void r_adc_interrupt(void)</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Clear the AWC bit of the ADM2 register and release the SNOOZE mode.</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Return Value</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td>None</td>
</tr>
</tbody>
</table>
5.6 Flowcharts

Figure 5.1 shows the entire flow for this application note.

![Flowchart]

**Figure 5.1 Overall Flowchart**
5.6.1 Main Processing

Figure 5.2, Figure 5.3, Figure 5.4, Figure 5.5 shows the flowchart for the main processing.

```
main()

Initialization setting function
hdwinit()

Main initializes settings
R_MAIN_UserInit()

Status transition (A)→(B)
R_MAIN_AtoB()

Status transition (B)→(E)
R_MAIN_BtoE()

Status transition (E)→(O)
R_MAIN_EtoO()

Status transition (O)→(E)
R_MAIN_OtoE()

Status transition (E)→(B)
R_MAIN_EtoB()

Status transition (E)→(D)
R_MAIN_EtoD()

Status transition (D)→(E)
R_MAIN_DtoE()

Status transition (E)→(M)
R_MAIN_DtoM()

A

After reset release (A), transition CPU to high-speed on-chip oscillator operation (B)

After CPU operates with high-speed on-chip oscillator clock(B), transition to sub clock operation (E)

Transition to HALT mode (O) while CPU is operating with sub clock(E)

Release HALT mode (O), then transition to sub clock operation (E)

Transition CPU from sub clock operation (E) to high-speed on-chip oscillator clock operation (B)

Transition CPU from high-speed on-chip oscillator clock operation (B) to high-speed system clock operation (D)

Transition CPU from high-speed system clock operation (D) to sub clock operation (E)

Transition CPU from sub clock operation (E) to high-speed clock operation (D)

Transition to HALT mode (M) while CPU is operating with high-speed system clock (D)
```
Figure 5.3  Main Processing (2/4)
Figure 5.4 Main Processing (3/4)

- **Status transition (C)→(D)**  
  \[ R\_\text{MAIN}\_\text{CtoD}(i) \]  
  Transition CPU from middle-speed on-chip oscillator clock operation (C) to high-speed system clock operation (D)

- **Status transition (D)→(F)**  
  \[ R\_\text{MAIN}\_\text{DtoF}(i) \]  
  Transition CPU from high-speed system clock operation (D) to low-speed on-chip oscillator clock operation (F)

- **Status transition (F)→(D)**  
  \[ R\_\text{MAIN}\_\text{FtoD}(i) \]  
  Transition CPU from low-speed on-chip oscillator clock operation (F) to high-speed system clock operation (D)

- **Status transition (D)→(C)**  
  \[ R\_\text{MAIN}\_\text{DtoC}(i) \]  
  Transition CPU from high-speed system clock operation (D) to middle-speed on-chip oscillator clock operation (C)

- **Status transition (C)→(J)**  
  \[ R\_\text{MAIN}\_\text{CtoJ}(i) \]  
  Transition to HALT mode (J) while CPU is operating with middle-speed on-chip oscillator clock operation (C)

- **Status transition (J)→(C)**  
  \[ R\_\text{MAIN}\_\text{JtoC}(i) \]  
  Release HALT mode (J), then transition to middle-speed on-chip oscillator clock operation (C)

- **Status transition (C)→(K)**  
  \[ R\_\text{MAIN}\_\text{CtoK}(i) \]  
  Transition to STOP mode (K) while CPU is operating with middle-speed on-chip oscillator clock operation (C)

- **Status transition (K)→(C)**  
  \[ R\_\text{MAIN}\_\text{KtoC}(i) \]  
  Release STOP mode (K), then transition to middle-speed on-chip oscillator clock operation (C)

- **Status transition (C)→(L)**  
  \[ R\_\text{MAIN}\_\text{CtoL}(i) \]  
  Transition to SNOOZE mode (L) while CPU is operating with middle-speed on-chip oscillator clock operation (C)

- **Status transition (L)→(C)**  
  \[ R\_\text{MAIN}\_\text{LtoC}(i) \]  
  Release SNOOZE mode (L), then transition to middle-speed on-chip oscillator clock operation (C)
Figure 5.5 Main Processing (4/4)

- Status transition (C)→(E)  
  R_MAIN_CtoE()  
  Transition CPU from middle-speed on-chip oscillator clock operation (C) to sub clock operation (E)

- Status transition (E)→(C)  
  R_MAIN_EtoC()  
  Transition CPU from sub clock operation (E) to middle-speed on-chip oscillator clock operation (C)

- Status transition (C)→(F)  
  R_MAIN_CtoF()  
  Transition CPU from middle-speed on-chip oscillator clock operation (C) to low-speed on-chip oscillator clock operation (F)

- Status transition (F)→(C)  
  R_MAIN_FtoC()  
  Transition CPU from low-speed on-chip oscillator clock operation (F) to middle-speed on-chip oscillator clock operation (C)

- Status transition (C)→(B)  
  R_MAIN_CtoB()  
  Transition CPU from middle-speed on-chip oscillator clock operation (C) to high-speed on-chip oscillator clock operation (B)

- Status transition (B)→(F)  
  R_MAIN_BtoF()  
  Transition CPU from high-speed on-chip oscillator clock operation (B) to low-speed on-chip oscillator clock operation (F)

- Status transition (F)→(P)  
  R_MAIN_FtoP()  
  Transition to HALT mode (P) while CPU is operating with low-speed on-chip clock oscillator (F)

- Status transition (P)→(F)  
  R_MAIN_PtoF()  
  Release HALT mode (P), then transition to low-speed on-chip oscillator clock operation (F)

- Status transition (F)→(B)  
  R_MAIN_FtoB()  
  Transition CPU from low-speed on-chip oscillator clock operation (F) to high-speed on-chip oscillator clock operation (B)

- End processing of status transition  
  R_MAIN_ENDX()  
  Set LED to OFF

- return
5.6.2 Initialization Function

Figure 5.6 shows the flowchart for the initialization function.

hdwinit()

Disable interrupts

System function
R_Systeminit()

return

IE ← 0

Figure 5.6 Initialization Function
5.6.3 System Function

Figure 5.7 shows the flowchart for system function.

![Flowchart for System Function]

**Figure 5.7 System Function**
5.6.4 Input/Output Port Settings

Figure 5.8 shows the flowchart for the input/output port settings.

```
R_PORT_Create(

Set port registers

Set port output mode registers

Set analog input
Alternate-function pins

Set port mode registers

return)

P2 ← 00000000B: Set P2.0 to low-level output
P5 ← 00000111B: Set P5.2 to P5.0 to high-level output
P6 ← 00000011B: Set P6.1 to P6.0 to high-level output
POM3 ← 00000000B: Set P3.0 to normal output mode
POM5 ← 00000000B: Set P5.6 to P5.1 to normal output mode

PM0 register ← 11100011B: Set P0.4 to P0.2 to digital input/output
PM1 register ← 00000000B: Set P1.7 to P1.0 to digital input/output
PM2 register ← 11000001B: Set P2.0 to analog input
PM3 register ← 11111100B: Set P3.1 to P3.0 to digital input/output
PM2 ← 11110001B: Set P2.0 to input ports
PM5 ← 00000000B: Set P5.2 to P5.0 to output ports
PM6 ← 11110000B: Set P6.1 to P6.0 to output ports
```

Figure 5.8 Input/Output Port Settings

Note: Refer to the initialization flowchart in the RL78/G13 Initialization (R01AN2575E) Application Note for details on how to set unused ports.

Caution: When designing circuits, always make sure unused ports are properly processed and all electrical characteristics are met. Also make sure each unused input-only port is connected to VDD or VSS through a resister.
5.6.5 Clock Generator Setting

Figure 5.9 shows the flowchart for setting the clock generator.

```
R_CGC_Create()

Set high-speed system clock/subsystem clock

CMC register ← 01010101B: X1 oscillation mode
XT oscillation mode
Ultra-low power consumption oscillation
X1 clock: 10 MHz ≤ fX ≤ 20 MHz

OSTC register ← 00000111B

MSTOP bit ← 1: X1 oscillator stopped

Set clock operation controls

MIOEN bit ← 0: Middle-speed on-chip oscillator stopped

Set Middle-speed on-chip oscillator clock

SELLOSC bit ← 0: Select sub clock as Subsystem clock

Set low-speed on-chip oscillator clock

XTSTOP bit ← 1: XT1 oscillator stopped

Set clock operation controls

Set subsystem clock supply mode control

OSMC register ← 10010000B: Provide fIL to 12-bit interval timer

Select CPU/peripheral hardware clock (fCLK)

CSS bit ← 0: Set main system clock (fMAIN) to CPU/peripheral hardware clock (fCLK)

MCM0 bit ← 0: Select main on-chip oscillator clock (fOCO) as main system clock (fMAIN)

MCM1 bit ← 0: Select high-speed on-chip clock (fIH) as main on-chip oscillator clock (fOCO)

HIOSTOP bit ← 0: High-speed on-chip oscillator operating

return
```

Figure 5.9 Clock Generator Setting
Clock operation mode setting

- Clock operation mode control register (CMC)
- High-speed system clock pin operation mode: X1 oscillation mode
- Subsystem clock pin operation mode: XT1 oscillation mode
- XT1 oscillator oscillation mode: ultra-low power consumption oscillation
- X1 clock oscillation frequency control: 10MHz < fX ≤ 20MHz

Symbol: CMC

<table>
<thead>
<tr>
<th>EXCLK</th>
<th>OSCSEL</th>
<th>EXCLKS</th>
<th>OSCSELS</th>
<th>AMPHS1</th>
<th>AMPHS0</th>
<th>AMPH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Bits 7-6

<table>
<thead>
<tr>
<th>EXCLK</th>
<th>OSCSEL</th>
<th>EXCLKS</th>
<th>OSCSELS</th>
<th>X1/P121 Port</th>
<th>X2/EXCLK/P122 Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Input port mode</td>
<td>Input port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>X1 oscillation mode</td>
<td>Crystal/ceramic resonator connection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Input port mode</td>
<td>Input port</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>External clock input mode</td>
<td>External clock input</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bits 5-4

<table>
<thead>
<tr>
<th>EXCLKS</th>
<th>OSCSELS</th>
<th>XT1/P123 Pin</th>
<th>XT2/EXCLKS/P124 Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Input port mode</td>
<td>Input port</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>XT1 oscillation mode</td>
<td>Crystal resonator connection</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Input port mode</td>
<td>Input port</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>External clock input mode</td>
<td>External clock input</td>
</tr>
</tbody>
</table>

Bits 2-1

<table>
<thead>
<tr>
<th>AMPHS1</th>
<th>AMPHS0</th>
<th>XT1 oscillator oscillation mode selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Low-power consumption oscillation (default)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Normal oscillation</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Ultra-low power consumption oscillation</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Setting prohibited</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>AMPH</th>
<th>Control of X1 clock oscillation frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1MHz ≤ fX ≤ 10MHz</td>
</tr>
<tr>
<td>1</td>
<td>10MHz &lt; fX ≤ 20MHz</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for details on how to set registers.
Operation control of clocks

- Clock operation status control register (CSC)
  - High-speed system clock operation control: X1 oscillator stopped
  - Subsystem clock operation control: XT1 oscillator stopped
  - Middle-speed on-chip oscillator clock operation control: Middle-speed on-chip oscillator stopped
  - High-speed on-chip oscillator clock operation control: High-speed on-chip oscillator operating

Symbol: CSC

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MSTOP</td>
<td>XTSTOP</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for details on how to set registers.
CPU/peripheral hardware clock \( (f_{\text{clk}}) \) setting

- System clock control register (CKC)
  - Status of \( f_{\text{clk}} \): main system clock
  - Selection of \( f_{\text{clk}} \): high-speed on-chip oscillator clock \( (f_{\text{H}}) \)

Symbol: CKC

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLS</td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</td>
<td>0</td>
<td>0</td>
<td>MCS1</td>
<td>MCM1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Bit 7

- **CLS**
  - Status of CPU/peripheral hardware clock \( (f_{\text{clk}}) \)
  - 0: Main system clock \( (f_{\text{MAIN}}) \)
  - 1: Subsystem clock \( (f_{\text{SUB}}) \)

Bit 6

- **CSS**
  - Selection of CPU/peripheral hardware clock \( (f_{\text{clk}}) \)
  - 0: Main system clock \( (f_{\text{MAIN}}) \)
  - 1: Subsystem clock \( (f_{\text{SUB}}) \)

Bit 5

- **MCS**
  - Status of main system clock \( (f_{\text{MAIN}}) \)
  - 0: Main on-chip oscillator clock \( (f_{\text{OCO}}) \)
  - 1: High-speed system clock \( (f_{\text{MX}}) \)

Bit 4

- **MCM0**
  - Main system clock \( (f_{\text{MAIN}}) \) operation control
  - 0: Selects main on-chip oscillator clock \( (f_{\text{OCO}}) \) as main system clock \( (f_{\text{MAIN}}) \)
  - 1: Selects high-speed system clock \( (f_{\text{MX}}) \) as main system clock \( (f_{\text{MAIN}}) \).

Bit 1

- **MCS1**
  - Status of main on-chip oscillator clock \( (f_{\text{SCO}}) \)
  - 0: High-speed on-chip oscillator clock \( (f_{\text{H}}) \)
  - 1: Middle-speed on-chip oscillator clock \( (f_{\text{M}}) \)

Bit 0

- **MCS1**
  - Main on-chip oscillator clock \( (f_{\text{SCO}}) \) operation control
  - 0: High-speed on-chip oscillator clock \( (f_{\text{H}}) \)
  - 1: Middle-speed on-chip oscillator clock \( (f_{\text{M}}) \)

Note: Refer to the RL78/I1D User’s Manual (Hardware) for details on how to set registers.
Subsystem clock supply mode control

- Subsystem clock supply mode control register (OSMC)
  Setting in STOP mode or in HALT mode while subsystem clock is selected as CPU clock
  : Stops supply of subsystem clock to peripheral functions other than the real-time clock 2, 12-bit interval timer,
    8-bit interval timer, and clock output/buzzer output controller.
  Selection of count clock for real-time clock and 12-bit interval timer: low-speed on-chip oscillator clock

  Symbol: OSMC

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTCLPC</td>
<td>0</td>
<td>0</td>
<td>WUTMMCK0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

  Bit 7

  RTCLPC Setting in STOP mode or in HALT mode while subsystem clock is selected as CPU clock
  0 Enables supply of subsystem clock to peripheral functions
  1 Stops supply of subsystem clock to peripheral functions other than the real-time clock 2, 12-bit interval timer, 8-bit interval timer, and clock output/buzzer output controller.

  Bit 4

  WUTMMCK0 Selection of count clock for real-time clock and 12-bit interval timer
  0 Subsystem clock (fSUB)
  1 Low-speed internal oscillator clock

Subsystem clock select

- Subsystem clock select register (CKSEL)
  Subsystem clock select: Select low-speed on-chip oscillator clock

  Symbol: CKSEL

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>SELLOSC</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

  Bit 0

  SELLOSC Selection of sub clock/low-speed on-chip oscillator clock
  0 Sub clock
  1 Low-speed on-chip oscillator clock

Note: Do not set SELLOSC to 1 when the sub clock (fSX, fSXR) operates.

Refer to the RL78/I1D User’s Manual (Hardware) for details on how to set registers.
5.6.6 External Interrupt Setting

Figure 5.10 shows the flowchart for setting the external interrupt.

```
R_INTC_Create()

Disable all INTP interrupt

MK0L register
PMK5-PMK0 bit ← 111111B : Interrupt processing disabled
MK0H register
PMK6 bit ← 1 : Interrupt processing disabled

IF0L register
PIF5-PIF0 bit ← 111111B : Interrupt request signal not generated
IF0H register
PIF6 bit ← 1 : Interrupt request signal not generated

Set interrupt priority level

PPR10 bit ← 1
PPR00 bit ← 1: Specify level 3 (low priority level)

Set external interrupt valid edge

EGP0 bit ← 0
EGN0 bit ← 1: Specify falling edge as INTP0 pin valid edge

return
```

**Figure 5.10 External Interrupt Setting**
Control of external interrupt valid edge

- External interrupt rising edge enable register (EGP0)
  Select valid edge for INTP0 pin: falling edge

Symbol: EGP0

<table>
<thead>
<tr>
<th></th>
<th>EGP6</th>
<th>EGP5</th>
<th>EGP4</th>
<th>EGP3</th>
<th>EGP2</th>
<th>EGP1</th>
<th>EGP0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
</tr>
</tbody>
</table>

Symbol: EGN0

<table>
<thead>
<tr>
<th></th>
<th>EGN6</th>
<th>EGN5</th>
<th>EGN4</th>
<th>EGN3</th>
<th>EGN2</th>
<th>EGN1</th>
<th>EGN0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>1</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>EGP0</th>
<th>EGN0</th>
<th>INTP0 pin valid edge selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Edge detection disabled</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Falling edge</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Rising edge</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Both rising and falling edges</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for details on how to set registers.
5.6.7 12-bit Interval Timer Setting

Figure 5.11 shows the flowchart for setting the 12-bit interval timer.

```
R_IT_Create()

Reset 12-bit interval timer
TMKARES bit ← 1
TMKARES bit ← 0

Supply clock to 12-bit interval timer
TMKAEN bit ← 1

Stop 12-bit interval timer operation
ITMC register ← 0000H

Disable 12-bit interval timer interrupt
TMKAMK bit ← 1

Clear 12-bit interval timer interrupt request flag
TMKAIF bit ← 0

Set interrupt priority level
TMKAPR1 bit ← 1
TMKAPR0 bit ← 1

Set the compare value (5ms) of 12-bit interval timer
ITMC register ← 004AH

return
```

**Figure 5.11 12-bit Interval Timer Setting**
12-bit interval timer clock supply setting
- Peripheral enable register 2 (PER2)
  Enable clock supply to 12-bit interval timer.

Symbol: PER2

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMKAEN</td>
<td>FMCEN</td>
<td>DOCEN</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Bit 7
- TMKAEN
  Control of 12-bit interval timer input clock supply
  - 0: Stops input clock supply.
  - 1: Enables input clock supply.

12-bit interval timer interval signal detection interrupt (INTIT) setting
- Interrupt request flag register (IF1H)
  Clear TMKAIF interrupt source flag.
- Interrupt mask flag register (MK1H)
  Set TMKAMK interrupt mask.

Symbol: IF1H

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DOCIF</td>
<td>CMPIF1</td>
<td>CMPIF0</td>
<td>KRIIF</td>
<td>TMKAIF</td>
<td>RTCIF</td>
<td>ADIF</td>
</tr>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Bit 2
- TMKAIF
  Interrupt request flag
  - 0: No interrupt request signal is generated
  - 1: Interrupt request signal is generated, interrupt request status

Symbol: MK1H

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DOCMK</td>
<td>CMPMK1</td>
<td>CMPMK0</td>
<td>KRMK</td>
<td>TMKAMK</td>
<td>RTCMK</td>
<td>ADMK</td>
</tr>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>1</td>
<td>X</td>
<td>x</td>
</tr>
</tbody>
</table>

Bit 2
- TMKAMK
  Interrupt servicing control
  - 0: Interrupt servicing enabled
  - 1: Interrupt servicing disabled

Note: Refer to the RL78/I1D User’s Manual (Hardware) for details on how to set registers.
12-bit interval timer interval signal detection interrupt (INTIT) setting

- Interval timer control register (ITMC)
  Start 12-bit interval timer count operation.

Symbol: ITMC

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RINTE</td>
<td>15</td>
<td>14</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ITCMP11-ITCMP0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>04AH</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bit 15

<table>
<thead>
<tr>
<th>RINTE</th>
<th>12-bit interval timer operation control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Count operation stopped (count clear)</td>
</tr>
<tr>
<td>1</td>
<td>Count operation started</td>
</tr>
</tbody>
</table>

Bits 11-0

<table>
<thead>
<tr>
<th>ITCMP11-ITCMP0</th>
<th>Specification of 12-bit interval timer compare value</th>
</tr>
</thead>
<tbody>
<tr>
<td>04AH</td>
<td>These bits generate an interrupt at the fixed cycle (count clock cycles x (ITCMP setting value 04AH + 1)).</td>
</tr>
<tr>
<td>000H</td>
<td>Setting prohibited</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for details on how to set registers.
5.6.8 Main initializes settings

Figure 5.12 shows the flowchart for the main initializes settings.

```
R_MAIN_UserInit()
|
| Enable interrupt IE ← 1
|
| Enable external interrupt PIF0 bit ← 0
|
| return PMK0 bit ← 0
```

**Figure 5.12 Main initializes settings**

Pin input edge detection interrupt (INTP0) setting

- Interrupt request flag register (IF0L)
  - Clear the PIF0 interrupt source flag.

- Interrupt mask flag register (MK0L)
  - Set PMK0 interrupt mask.

Symbol: IF0L

```
7 6 5 4 3 2 1 0
PIF5 PIF4 PIF3 PIF2 PIF1 PIF0 LVIF WDTIF
```

Bit 2

<table>
<thead>
<tr>
<th>PIF0</th>
<th>Interrupt request flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Interrupt request signal is generated, interrupt request status</td>
</tr>
<tr>
<td>0</td>
<td>No interrupt request signal is generated</td>
</tr>
</tbody>
</table>

Symbol: MK0L

```
7 6 5 4 3 2 1 0
PMK5 PMK4 PMK3 PMK2 PMK1 PMK0 LVIMK WDTMK
```

Bit 2

<table>
<thead>
<tr>
<th>PMK0</th>
<th>Interrupt servicing control</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Interrupt servicing disabled</td>
</tr>
<tr>
<td>0</td>
<td>Interrupt servicing enabled</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for details on how to set registers.
5.6.9 Status Transition AtoB

Figure 5.13 shows the flowchart for status transition AtoB.

![Flowchart for Status Transition AtoB](image)

5.6.10 CPU operation (NOP instruction execution)

Figure 5.14 shows the flowchart for the CPU operation (NOP instruction execution)

![Flowchart for CPU Operation (NOP instruction execution)](image)
5.6.11 Status Transition BtoE

Figure 5.15 and Figure 5.16 show the flowchart for status transition BtoE.

---

**Figure 5.15 Status Transition BtoE (1/2)**
Start count of 12-bit interval timer

ITMC register ← 8001H

CLS bit = 1?

No

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

To status transition error processing

R_MAIN_ERROR()

Yes

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Stop high-speed on-chip oscillator

HIOSTOP bit ← 1

Control LED lighting (sub clock)

P5.0 bit ← 0: Set LED3 to OFF
P5.1 bit ← 0: Set LED4 to OFF
P5.2 bit ← 1: Set LED5 to ON

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

Execute NOP instruction continuously.

return

Figure 5.16 Status Transition BtoE (2/2)
5.6.12 Error Processing of Status Transition

Figure 5.17 shows the flowchart for error processing of status transition.

![Flowchart for error processing of status transition](image)

5.6.13 Status Transition EtoO

Figure 5.18 shows the flowchart for status transition EtoO.

![Flowchart for status transition EtoO](image)
5.6.14 Status Transition OtoE

Figure 5.19 shows the flowchart for status transition OtoE.

Figure 5.19 Status Transition OtoE
5.6.15 Status Transition EtoB

Figure 5.20 and Figure 5.21 show the flowchart for status transition EtoB.

![Flowchart for status transition EtoB](image)

Figure 5.20 Status Transition EtoB(1/2)
Start 12-bit interval timer count

CLS bit = 0?

Yes

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

ITM register ← 8001H

R_MAIN_ERROR()

To status transition error processing

No

TMKAMK bit ← 1

ITM register ← 0000H

XTSTOP bit ← 0

Disable interval detection interrupt

Stop 12-bit interval timer count

Stop XT1 oscillator

Control LED lighting (high-speed on-chip oscillator clock)

P5.0 bit ← 1: Set LED3 to OFF
P5.1 bit ← 1: Set LED4 to OFF
P5.2 bit ← 0: Set LED5 to ON

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

execute NOP instruction continuously

return

Figure 5.21 Status Transition EtoB(2/2)
5.6.16 Status Transition BtoD
Figure 5.22 and Figure 5.23 shows the flowchart for status transition BtoD.

**Diagram: R_MAIN_BtoD()**

1. Disable external interrupt
2. Set oscillation stabilization time selection register
3. X1 oscillator operation
4. Oscillation stability time elapsed?
   - No
   - Yes (OSCT register branches at FFH)
     - Select high-speed system clock as main system clock
     - Clear interval detection interrupt request flag
     - Enable interval detection interrupt
     - Initialize interval detection interrupt generation counter

**Figure 5.22 Status Transition BtoD(1/2)**

- PMK0 bit ← 1
- OSTS register ← 07H
- MSTOP bit ← 0
- MCM0 bit ← 1
- TMKAIF bit ← 0
- TMKAMK bit ← 0
Start count of 12-bit interval timer

MCS bit = 1?

Yes

Interval detection interrupt generated time?

No

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR()

To status transition error processing

Disable interval detection interrupt

ITMC register ← 8001H

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

HIOSTOP bit ← 1

Stop high-speed on-chip oscillator

Control LED lighting (high-speed system clock)

P5.0 bit ← 0: Set LED5 to ON

P5.1 bit ← 1: Set LED4 to OFF

P5.2 bit ← 0: Set LED5 to ON

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON

P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)

R_MAIN_NOP_Loop()

Execute NOP instruction continuously.

return

Figure 5.23 Status Transition BtoD(2/2)
5.6.17 Status Transition DtoE

Figure 5.24 and Figure 5.25 shows the flowchart for status transition DtoE.

![Flowchart](image)

**Figure 5.24 Status Transition DtoE(1/2)**
Start count of 12-bit interval timer

CLS bit = 1?

No

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

ITMC register ← 8001H

TMKAMK bit ← 1

ITMC register ← 0000H

MSTOP bit ← 1

P5.0 bit ← 0: Set LED3 to ON
P5.1 bit ← 0: Set LED4 to ON
P5.2 bit ← 1: Set LED5 to OFF

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

PIF0 bit ← 0

PMK0 bit ← 0

Execute NOP instruction continuously.

Control LED lighting (sub clock)

P5.0 bit ← 0: Set LED3 to ON
P5.1 bit ← 0: Set LED4 to ON
P5.2 bit ← 1: Set LED5 to OFF

Control LED lighting (CPU operation)

Clear external interrupt request flag

Enable external interrupt servicing

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

return

Figure 5.25 Status Transition DtoE(2/2)
5.6.18 Status Transition EtoD

Figure 5.26 and Figure 5.27 shows the flowchart for status transition EtoD.

```
R_MAIN_EtoD()

Disable external interrupt

Set oscillation stabilization time selection register

X1 oscillator operation

No

Oscillation stability time elapsed?

Yes (OSCT register branches at FFH)

Select high-speed system clock as CPU/peripheral hardware clock

Clear interval detection interrupt request flag

Enable interval detection interrupt

Initialize interval detection interrupt generation counter

PMK0 bit ← 1

OSTS register ← 07H

MSTOP bit ← 0

CSS bit ← 0

TMKAIF bit ← 0

TMKAMK bit ← 0
```

Figure 5.26 Status Transition EtoD(1/2)
Figure 5.27  Status Transition EtoD(2/2)

A

Start count of 12-bit interval timer

ITMC register ← 8001H

CLS bit = 0?

No

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR()

No

To status transition error processing

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Stop XT1 oscillator

XTSTOP bit ← 1

Control LED lighting (high-speed system clock)

P5.0 bit ← 0: Set LED3 to ON
P5.1 bit ← 1: Set LED4 to OFF
P5.2 bit ← 0: Set LED5 to ON

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

Execute NOP instruction continuously.

return
5.6.19 Status Transition DtoM

Figure 5.18 shows the flowchart for status transition DtoM.

```
  R_MAIN_DtoM()
    Control LED lighting (HALT mode)
      P6.0 bit ← 1: Set LED1 to OFF
      P6.1 bit ← 0: Set LED2 to ON
    Clear external interrupt request flag
      PIF0 bit ← 0
    HALT
    return
```

Figure 5.18 Status Transition DtoM

5.6.20 Status Transition MtoD

Figure 5.18 shows the flowchart for status transition MtoD.

```
  R_MAIN_MtoD()
    Control LED lighting (CPU operation)
      P6.0 bit ← 0: Set LED1 to ON
      P6.1 bit ← 0: Set LED2 to ON
    Clear external interrupt request flag
      PIF0 bit ← 0
    CPU operation (NOP instruction execution)
      Execute NOP instruction continuously.
      R_MAIN_NOP_Loop()
    return
```

Figure 5.18 Status Transition MtoD
5.6.21 Status Transition DtoN
Figure 5.30 shows the flowchart for status transition DtoN.

Figure 5.30 Status Transition DtoN

5.6.22 Status Transition NtoD
Figure 5.31 shows the flowchart for status transition NtoD.

Figure 5.31 Status Transition NtoD
5.6.23 Status TransitionDtoB

Figure 5.32 and Figure 5.33 shows the flowchart for status transition DtoB.

![Flowchart for Status Transition DtoB](image_url)

**Figure 5.32 Status Transition DtoB (1/2)**
Figure 5.33 Status Transition DtoB(2/2)
5.6.24 Status Transition BtoI
Figure 5.34 shows the flowchart for status transition BtoI.

![Flowchart of Status Transition BtoI](image)

- **R_MAIN_BtoI()**
  - Control LED lighting (HALT mode)
    - P6.0 bit ← 1: Set LED1 to OFF
    - P6.1 bit ← 0: Set LED2 to ON
  - Clear external interrupt request flag
  - HALT
  - return

**Figure 5.34** Status Transition BtoI

5.6.25 Status Transition ItoB
Figure 5.35 shows the flowchart for status transition ItoB.

![Flowchart of Status Transition ItoB](image)

- **R_MAIN_ItoB()**
  - Control LED lighting (CPU operation)
    - P6.0 bit ← 0: Set LED1 to ON
    - P6.1 bit ← 0: Set LED2 to ON
  - Clear external interrupt request flag
    - PIF0 bit ← 0
  - CPU operation (NOP instruction execution)
    - Execute NOP instruction continuously.
    - R_MAIN_NOP_Loop()
  - return

**Figure 5.35** Status Transition ItoB
5.6.26 Status Transition BtoG
Figure 5.36 shows the flowchart for status transition BtoG.

```
R_MAIN_BtoG()

Control LED lighting (STOP mode)
P6.0 bit ← 1: Set LED1 to OFF
P6.1 bit ← 1: Set LED2 to OFF

Clear external interrupt request flag
PIF0 bit ← 0

STOP mode

return
```

Figure 5.36 Status Transition BtoG

5.6.27 Status Transition GtoB
Figure 5.37 shows the flowchart for status transition GtoB.

```
R_MAIN_GtoB()

Control LED lighting (CPU operation)
P6.0 bit ← 0; Set LED1 to ON
P6.1 bit ← 0; Set LED2 to ON

Clear external interrupt request flag
PIF0 bit ← 0

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

Execute NOP instruction continuously.

return
```

Figure 5.37 Status Transition GtoB
5.6.28 Status Transition BtoH

Figure 5.38 shows the flowchart for status transition BtoH.

```
R_MAIN_BtoH()
  Disable external interrupt
    PMK0 bit ← 1
  Set A/D SNOOZE mode
    R_MAIN_AD_SnoozeOn()
    Set A/D conversion execution to hardware trigger
    with the 12-bit interval timer interrupt signal
  Disable interval detection interrupt
    TMKAMK bit ← 1
  Control LED lighting
    (SNOOZE mode)
    P6.0 bit ← 0: Set LED1 to ON
    P6.1 bit ← 1: Set LED2 to OFF
  Start 12-bit interval timer count
    ITMC register ← 8FFH
  STOP
  return
```

Figure 5.38 Status Transition BtoH
5.6.29 A/D Converter Setting

Figure 5.39 shows the flowchart for setting the A/D converter.

![Flowchart](image)

**Figure 5.39  A/D Converter Setting**

SNOOZE mode setting

- A/D converter mode register 2 (ADM2)
  - Set SNOOZE mode.

Symbol: ADM2

<table>
<thead>
<tr>
<th>ADREFP1</th>
<th>ADREFP0</th>
<th>ADREFM</th>
<th>0</th>
<th>ADCRK</th>
<th>AWC</th>
<th>0</th>
<th>ADTYP</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>x</td>
</tr>
</tbody>
</table>

Bit 2

<table>
<thead>
<tr>
<th>AWC</th>
<th>Specification of SNOOZE mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Do not use the SNOOZE mode function</td>
</tr>
<tr>
<td>1</td>
<td>Use the SNOOZE mode function</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for detailed explanations on how to set registers.
5.6.30 A/D Converter Initial Setting

Figure 5.40 shows the flowchart for setting the A/D converter.

Figure 5.29 A/D Converter Initial Setting
A/D conversion time and operation mode settings

- A/D converter mode register 0 (ADM0)
  - Control the A/D conversion operation.
  - Specify the A/D conversion channel selection mode.

Symbol: ADM0

<table>
<thead>
<tr>
<th>ADCS</th>
<th>ADM0</th>
<th>FR2</th>
<th>FR1</th>
<th>FR0</th>
<th>LV1</th>
<th>LV0</th>
<th>ADCE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Bit 6

<table>
<thead>
<tr>
<th>ADM0</th>
<th>Specification of A/D channel selection mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Select mode</td>
</tr>
<tr>
<td>1</td>
<td>Scan mode</td>
</tr>
</tbody>
</table>

Bits 5-1

<table>
<thead>
<tr>
<th>ADM0</th>
<th>Mode</th>
<th>Conversion Time</th>
<th>Conversion clock (fCLK)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>fCLK 1MHz</td>
<td>fCLK 4MHz</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>Setting prohibited</td>
<td>Setting prohibited</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>54.25 μs</td>
<td>36.1667 μs</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>54.25 μs</td>
<td>27.25 μs</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>41 μs</td>
<td>20.5 μs</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>34.25 μs</td>
<td>17.125 μs</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>55 μs</td>
<td>27.5 μs</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>28 μs</td>
<td>14 μs</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>56 μs</td>
<td>28 μs</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>ADCE</th>
<th>A/D voltage comparator operation control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Stops A/D enables comparator operation</td>
</tr>
<tr>
<td>1</td>
<td>Enables A/D voltage comparator operation</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for detailed explanations on how to set registers.
A/D conversion trigger mode setting

- A/D converter mode register 1 (ADM1)
  Select the A/D conversion trigger mode.
  Specify the A/D conversion operation mode.

Symbol: ADM1

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>ADTMD1</td>
<td>ADTMD0</td>
<td>ADSCM</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ADTRS1</td>
<td>ADTRS0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Bits 7-6

<table>
<thead>
<tr>
<th>ADTMD1</th>
<th>ADTMD0</th>
<th>Selection of the A/D conversion trigger mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>—</td>
<td>Software trigger mode</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Hardware trigger no-wait mode</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Hardware trigger wait mode</td>
</tr>
</tbody>
</table>

Bit 5

<table>
<thead>
<tr>
<th>ADSCM</th>
<th>Specification of the A/D conversion mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Sequential conversion mode</td>
</tr>
<tr>
<td>1</td>
<td>One-shot conversion mode</td>
</tr>
</tbody>
</table>

Bits 1-0

<table>
<thead>
<tr>
<th>ADTRS1</th>
<th>ADTRS0</th>
<th>Selection of the hardware trigger signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>End of timer channel 01 count or capture interrupt signal (INTTM01)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Event signal selected by ELC</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Setting prohibited</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>12-bit interval timer interrupt signal (INTIT)</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for detailed explanations on how to set registers.
Reference voltage source setting

- A/D converter mode register 2 (ADM2)
  Set the reference voltage source.

Symbol: ADM2

<table>
<thead>
<tr>
<th>Bit 7-6</th>
<th>Selection of the + side reference voltage source of the A/D converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADREFP1</td>
<td>ADREFP0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Bit 5

<table>
<thead>
<tr>
<th>Bit 5</th>
<th>Selection of the - side reference voltage source of the A/D converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADREFM</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Supplied from AVSS</td>
</tr>
<tr>
<td>1</td>
<td>Supplied from P21/AVREFM/ANI1</td>
</tr>
</tbody>
</table>

Bit 3

<table>
<thead>
<tr>
<th>Bit 3</th>
<th>Checking the upper and lower limit conversion result values</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADCRK</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>The A/D conversion end interrupt request signal (INTAD) is output when the ADLL register &lt; the ADCR register or the ADUL register &lt; the ADCR register (AREA2) or the ADUL register &lt; the ADCR register (AREA3).</td>
</tr>
<tr>
<td>1</td>
<td>The A/D conversion end interrupt request signal (INTAD) is output when the ADLR register &lt; the ADLL register (AREA1).</td>
</tr>
</tbody>
</table>

Bit 2

<table>
<thead>
<tr>
<th>Bit 2</th>
<th>Specification of the SNOOZE mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWC</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Do not use the SNOOZE mode function.</td>
</tr>
<tr>
<td>1</td>
<td>Use the SNOOZE mode function.</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>Bit 0</th>
<th>Selection of the A/D conversion resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADTYP</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>12-bit resolution</td>
</tr>
<tr>
<td>1</td>
<td>8-bit resolution</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/I1D User’s Manual (Hardware) for detailed explanations on how to set registers.
Conversion result comparison upper/lower limit settings

- Conversion result comparison upper limit setting register (ADUL)
- Conversion result comparison lower limit setting register (ADLL)

Set conversion result comparison upper/lower limit values.

Symbol: ADUL

<table>
<thead>
<tr>
<th>ADUL7</th>
<th>ADUL6</th>
<th>ADUL5</th>
<th>ADUL4</th>
<th>ADUL3</th>
<th>ADUL2</th>
<th>ADUL1</th>
<th>ADUL0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Symbol: ADLL

<table>
<thead>
<tr>
<th>ADLL7</th>
<th>ADLL6</th>
<th>ADLL5</th>
<th>ADLL4</th>
<th>ADLL3</th>
<th>ADLL2</th>
<th>ADLL1</th>
<th>ADLL0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Input channel specification

- Analog input channel specification register (ADS)

Specify the input channel of the analog voltage to be A/D converted.

Symbol: ADS

<table>
<thead>
<tr>
<th>ADISS</th>
<th>ADS4</th>
<th>ADS3</th>
<th>ADS2</th>
<th>ADS1</th>
<th>ADS0</th>
<th>Analog input channel</th>
<th>Input source</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ANI0</td>
<td>P10/ANI0 /AVREF pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>ANI1</td>
<td>P11/ANI1 /AVREF pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>ANI2</td>
<td>P12/ANI2 pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>ANI3</td>
<td>P13/ANI3 pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ANI4</td>
<td>P14/ANI4 pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>ANI5</td>
<td>P15/ANI5 pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>ANI6</td>
<td>P16/ANI6 pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>ANI7</td>
<td>P17/ANI7 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ANI8</td>
<td>P25/ANI8 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>ANI9</td>
<td>P24/ANI9 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>ANI10</td>
<td>P23/ANI10 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ANI11</td>
<td>P21/ANI10 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>ANI12</td>
<td>P21/ANI12 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>ANI13</td>
<td>P20/ANI13 pin</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ANI16</td>
<td>P02/ANI16 pin</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>ANI17</td>
<td>P03/ANI17 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>ANI18</td>
<td>P04/ANI18 pin</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>–</td>
<td>Temperature sensor 0 output</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>–</td>
<td>Internal reference voltage output (1.45 V)</td>
</tr>
</tbody>
</table>

Other than the above Setting prohibited
5.6.31 Status Transition HtoB

Figure 5.41 shows the status transition HtoB.

![Diagram of status transition HtoB]

- **Stop A/D voltage converter operation**
  - ADCE bit ← 0

- **Stop the clock to A/D converter**
  - ADCEN bit ← 0

- **Disable A/D conversion complete interrupt**
  - ADMK bit ← 1

- **Stop 12-bit interval timer count**
  - ITMC register ← 0000H

- **Control LED lighting (CPU operation)**
  - P6.0 bit ← 0: Set LED1 to ON
  - P6.1 bit ← 0: Set LED2 to ON

- **Clear external interrupt request flag**
  - PIF0 bit ← 0

- **Enable external interrupt servicing**
  - PMK0 bit ← 0

- **CPU operation (NOP instruction execution)**
  - Execute NOP instruction continuously.

**Figure 5.41 Status Transition HtoB**
5.6.32 Status Transition BtoC

Figure 5.42 and Figure 5.43 shows the status transition BtoC.

![Diagram](image)

**Figure 5.42 Status Transition BtoC(1/2)**
Start count of 12-bit interval timer

MCS1 bit = 1?

Yes

ITMC register ← 8001H

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR() To status transition error processing

No

No

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Stop high-speed on-chip oscillator

HIOSTOP bit ← 1

Control LED lighting (middle-speed on-chip oscillator clock)

P5.0 bit ← 1: Set LED3 to OFF
P5.1 bit ← 0: Set LED4 to ON
P5.2 bit ← 1: Set LED5 to OFF

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)

R_MAIN_NOP_Loop()

Execute NOP instruction continuously

return

Figure 5.43 Status Transition BtoC(2/2)
5.6.33 Status Transition CtoD

Figure 5.44 and Figure 5.45 shows the status transition CtoD.

```
R_MAIN_CtoD()

<table>
<thead>
<tr>
<th>Disable external interrupt</th>
<th>PMK0 bit ← 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set oscillation stabilization</td>
<td>OSTS register ← 07H</td>
</tr>
<tr>
<td>time selection register</td>
<td>MSTOP bit ← 0</td>
</tr>
<tr>
<td>X1 oscillator operation</td>
<td></td>
</tr>
<tr>
<td>Oscillation stability</td>
<td></td>
</tr>
<tr>
<td>time elapsed?</td>
<td></td>
</tr>
<tr>
<td>Yes (OSCT register branches at FFH)</td>
<td></td>
</tr>
<tr>
<td>Select high-speed system clock as</td>
<td>MCM0 bit ← 1</td>
</tr>
<tr>
<td>CPU/peripheral hardware clock</td>
<td></td>
</tr>
<tr>
<td>Clear interval detection interrupt</td>
<td>TMKAIF bit ← 0</td>
</tr>
<tr>
<td>interrupt request flag</td>
<td></td>
</tr>
<tr>
<td>Enable interval detection</td>
<td>TMKAMK bit ← 0</td>
</tr>
<tr>
<td>interrupt</td>
<td></td>
</tr>
<tr>
<td>Initialize interval detection</td>
<td></td>
</tr>
<tr>
<td>interrupt generation counter</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5.44 Status Transition CtoD(1/2)
Figure 5.45 Status Transition CtoD(2/2)

Start count of 12-bit interval timer

ITMC register ← 8001H

MCS bit = 1?

Yes

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR()

To status transition error processing

No

No

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Middle-speed on-chip oscillator stopped

MIOEN bit ← 0

Control LED lighting (high-speed system clock)

P5.0 bit ← 0: Set LED3 to ON
P5.1 bit ← 1: Set LED4 to OFF
P5.2 bit ← 0: Set LED5 to ON

Control LED lighting (CPU operation)

P6.1 bit ← 0: Set LED1 to ON
P6.0 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)
MAIN_NOP_Loop()

Execute NOP instruction continuously

return
5.6.34 Status Transition DtoF

Figure 5.46 and Figure 5.47 shows the status transition DtoF.

```
R_MAIN_DtoF()

Disable external interrupt
PMK0 bit ← 1

Select low-speed on-chip oscillator clock as sub system clock
SELLOSC bit ← 1

Select low-speed on-chip oscillator clock as CPU/peripheral hardware clock
CSS bit ← 1

Clear interval detection interrupt request flag
TMKAIF bit ← 0

Enable interval detection interrupt
TMKAMK bit ← 0

Initialize interval detection interrupt generation counter
ITMC register ← 8001H

Start 12-bit interval timer count

CLS bit = 1?

No

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()

To status transition error processing

No

A

Figure 5.46 Status Transition DtoF(1/2)```
Figure 5.47  Status Transition DtoF(2/2)
5.6.35 Status Transition FtoD

Figure 5.48 and Figure 5.49 shows the status transition FtoD.

\[
\begin{align*}
\text{R_MAIN_FtoD()} & \quad \text{PMK0 bit} \leftarrow 1 \\
\text{Set oscillation stabilization time selection register} & \quad \text{OSTS register} \leftarrow 07H \\
\text{X1 oscillator operation} & \quad \text{MSTOP bit} \leftarrow 0 \\
\text{Oscillation stabilization time elapsed?} & \\
\text{Yes (branch when OSCT register is FFH)} & \quad \text{CSS bit} \leftarrow 0 \\
\text{Select high-speed system clock as CPU/peripheral hardware clock} & \quad \text{TMKAIF bit} \leftarrow 0 \\
\text{Clear interval detection interrupt request flag} & \quad \text{TMKAMK bit} \leftarrow 0 \\
\text{Enable interval detection interrupt} & \\
\text{Initialize interval detection interrupt generation counter} & \\
\end{align*}
\]

Figure 5.48 Status Transition FtoD(1/2)
Start count of 12-bit interval timer

ITMC register ← 8005H

CLS bit = 0?

No

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR()

To status transition error processing

Yes

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Select sub clock as subsystem clock

SELOSC bit ← 0

Control LED lighting (high-speed system clock)

P5.0 bit ← 0: Set LED3 to ON

P5.1 bit ← 1: Set LED4 to OFF

P5.2 bit ← 0: Set LED5 to ON

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON

P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)

R_MAIN_NOP_Loop()

Execute NOP instruction continuously

return

Figure 5.49 Status Transition FtoD(2/2)
5.6.36 Status Transition DtoC

Figure 5.50 and Figure 5.51 shows the status transition DtoC.

![Status Transition Diagram]

Figure 5.50 Status Transition DtoC(1/2)
Start count of 12-bit interval timer

MCS bit = 0?

Yes

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR() To status transition error processing

No

ITMC register ← 8001H

ITMC register ← 0000H

TMKAMK bit ← 1

MSTOP bit ← 1

Disable interval detection interrupt

Stop count of 12-bit interval timer

Stop X1 oscillator

Control LED lighting (middle-speed on-chip oscillator clock)

Control LED lighting (CPU operation)

Clear external interrupt request flag

Enable external interrupt servicing

CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()

return

Figure 5.51 Status Transition DtoC(2/2)
5.6.37 Status Transition CtoJ
Figure 5.52 shows the status transition CtoJ.

```
R_MAIN_CtoJ()

Control LED lighting (HALT mode)

Clear external interrupt request flag

HALT

return
```

**Figure 5.52 Status Transition CtoJ**

5.6.38 Status Transition JtoC
Figure 5.53 shows the status transition JtoC.

```
R_MAIN_JtoC()

Control LED lighting (CPU operation)

Clear external interrupt request flag

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

return
```

**Figure 5.53 Status Transition JtoC**
5.6.39 Status Transition CtoK

Figure 5.54 shows the status transition CtoK.

```c
R_MAIN_CtoK()

Control LED lighting (STOP mode)
P6.0 bit ← 1: Set LED1 to OFF
P6.1 bit ← 1: Set LED2 to OFF

Clear external interrupt request flag

STOP mode

return
```

**Figure 5.54 Status Transition CtoK**

5.6.40 Status Transition KtoC

Figure 5.55 shows the status transition KtoC.

```c
R_MAIN_KtoC()

Control LED lighting (CPU operation)
P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

CPU operation (NOP instruction execution)

_NOP

return
```

**Figure 5.55 Status Transition KtoC**
5.6.41 Status Transition CtoL

Figure 5.56 shows the status transition CtoL.

```
R_MAIN_CtoL()

Disable external interrupt

Set A/D SNOOZE mode
R_MAIN_AD_SnoozeOn()

Disable interval detection interrupt

Control LED lighting (SNOOZE mode)

Start 12-bit interval timer count

STOP

return
```

PM[0] bit ← 1

Set A/D conversion execution to hardware trigger with the 12-bit interval timer interrupt signal

TMKAMK bit ← 1

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 1: Set LED2 to OFF

ITMC register ← 8FFFH

**Figure 5.56** Status Transition CtoL
5.6.42 Status Transition LtoC

Figure 5.57 shows the status transition LtoC.

```c
R_MAIN_LtoC()

Stop A/D voltage converter operation

Stop the clock to A/D converter

Disable A/D conversion complete interrupt

ADCE bit ← 0

ADCN bit ← 0

ADMK bit ← 1

Stop 12-bit interval timer count

ITMC register ← 0000H

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON

P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

Execute NOP instruction continuously.

return
```

**Figure 5.57** Status Transition LtoC
5.6.43 Status Transition CtoE

Figure 5.58 and Figure 5.59 show the status transition CtoE.

Figure 5.58 Status Transition CtoE(1/2)
Start count of 12-bit interval timer

ITMC register ← 8001H

CLS bit = 1?

Yes

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR() To status transition error processing

No

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Stop middle-speed on-chip oscillator

MIOEN bit ← 1

Control LED lighting (sub clock)

P5.0 bit ← 0: Set LED3 to ON

P5.1 bit ← 0: Set LED4 to ON

P5.2 bit ← 1: Set LED5 to OFF

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON

P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)

R_MAIN_NOP_Loop()

Execute NOP instruction continuously.

return

Figure 5.59 Status Transition CtoE(2/2)
5.6.44 Status Transition EtoC

Figure 5.60 and Figure 5.61 shows the status transition EtoC.

Figure 5.60  Status Transition EtoC(1/2)
Start count of 12-bit interval timer

ITMC register ← 8001H

CLS bit = 0?

No

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR()

To status transition error processing

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Stop XT1 oscillator

XTSTOP bit ← 1

Control LED lighting (middle-speed on-chip oscillator clock)

P5.0 bit ← 1: Set LED3 to OFF
P5.1 bit ← 0: Set LED4 to ON
P5.2 bit ← 1: Set LED5 to OFF

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)

R_MAIN_NOP_Loop()

Execute NOP instruction continuously

return

Figure 5.61 Status Transition EtoC(2/2)
Figure 5.62 and Figure 5.63 shows the status transition CtoF.

R_MAIN_CtoF()

Disable external interrupt

PMK0 bit ← 1

Select low-speed on-chip oscillator clock as sub system clock

SELLOSC bit ← 1

Select low-speed on-chip oscillator clock as CPU/peripheral hardware clock

CSS bit ← 1

Clear interval detection interrupt request flag

TMKAIF bit ← 0

Enable interval detection interrupt

TMKAMK bit ← 0

Initialize interval detection interrupt generation counter

ITMC register ← 8001H

Start 12-bit interval timer count

CLS bit = 1?

No

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

To status transition error processing

R_MAIN_ERROR()

A

Figure 5.62 Status Transition CtoF(1/2)
Figure 5.63 Status Transition CtoF(2/2)
5.6.46 Status Transition FtoC

Figure 5.64 and Figure 5.65 shows the status transition FtoC.

```
R_MAIN_FtoC()

Disable external interrupt

Selection of middle-speed on-chip oscillator clock frequency

Middle-speed on-chip oscillator operation

Clear interval detection interrupt request flag

Enable interval detection interrupt

Initialize interval detection interrupt generation counter

Start count of 12-bit interval timer

ITMC register ← 8001H

Interval detection interrupt generated 1 time?

Yes (branch when g_int_cnt = 1)

Stop count of 12-bit interval timer

ITMC register ← 0000H

Select middle-speed on-chip oscillator clock as CPU / peripheral hardware clock

Clear interval detection interrupt request flag

TMKAIF bit ← 0

Initialize interval detection interrupt generation counter

PMK0 bit ← 1

MOCODIV register ← 00H: Set the middle-speed on-chip oscillator frequency to 4MHz

MIOEN bit ← 1

TMKAIF bit ← 0

TMKAMK bit ← 0

CSS bit ← 0
```

Figure 5.64 Status Transition FtoC(1/2)
Figure 5.65  Status Transition FtoC(2/2)

A

Start count of 12-bit interval timer

ITMC register ← 8005H

CLS bit = 0?

No

Interval detection interrupt generated time?

No

YES (branch when g_int_cnt = 1)

R_MAIN_ERROR()

To status transition error processing

Yes

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Select sub clock as subsystem clock

SELOSC bit ← 0

Control LED lighting (middle-speed on-chip oscillator clock)

P5.0 bit ← 1: Set LED3 to OFF
P5.1 bit ← 0: Set LED4 to ON
P5.2 bit ← 1: Set LED5 to OFF

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED1 to ON

Clear external interrupt request request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)

R_MAIN_NOP_LOOP()

Execute NOP instruction continuously

return
5.6.47 Status Transition CtoB

Figure 5.66 and Figure 5.67 shows the status transition CtoB.

![Flowchart showing the status transition CtoB](image)

Figure 5.66 Status Transition CtoB(1/2)
Start 12-bit interval timer count

MCS1 bit = 0?

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR() To status transition error processing

Disable interval detection interrupt

TMKAMK bit ← 1

Stop 12-bit interval timer count

ITMC register ← 0000H

Stop middle-speed on-chip oscillator

MIOEN bit ← 0

Control LED lighting (high-speed on-chip oscillator clock)

P5.0 bit ← 1: Set LED4 to OFF
P5.1 bit ← 1: Set LED4 to OFF
P5.2 bit ← 0: Set LED5 to ON

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)

Execute NOP instruction continuously

return

Figure 5.67 Status Transition CtoB(2/2)
5.6.48 Status Transition BtoF
Figure 5.68 and Figure 5.69 shows the status transition BtoF.

![Flowchart of Status Transition BtoF](image)

Figure 5.68 Status Transition BtoF(1/2)
Disable interval detection interrupt

Stop 12-bit interval timer count

Stop high-speed on-chip oscillator

Control LED lighting (low-speed on-chip oscillator clock)

Control LED lighting (CPU operation)

Clear external interrupt request flag

Enable external interrupt servicing

CPU operation (NOP instruction execution)
RMAIN_NOP_Loop()

return

TMKAMK bit ← 1

ITMC register ← 0000H

HIOSTOP bit ← 1

P5.0 bit ← 1: Set LED3 to OFF
P5.1 bit ← 0: Set LED4 to ON
P5.2 bit ← 0: Set LED5 to ON

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

PIF0 bit ← 0

PMK0 bit ← 0

Execute NOP instruction continuously

Figure 5.69 Status Transition BtoF(2/2)
5.6.49 Status Transition FtoP

Figure 5.70 shows the status transition FtoP.

```
R_MAIN_FtoP()

Control LED lighting (HALT mode)

P6.0 bit ← 1: Set LED1 to OFF
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

HALT

return
```

Figure 5.70 Status Transition FtoP

5.6.50 Status Transition PtoF

Figure 5.71 shows the status transition PtoF.

```
R_MAIN_PtoF()

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request flag

PIF0 bit ← 0

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

Execute NOP instruction continuously

return
```

Figure 5.71 Status Transition PtoF
5.6.51 Status Transition FtoB

Figure 5.72 and Figure 5.73 shows the status transition FtoB.

![Diagram](image)

**Figure 5.72 Status Transition FtoB(1/2)**
Start count of 12-bit interval timer

ITMC register ← 8005H

CLS bit = 0?

No

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR()

To status transition error processing

Yes

Disable interval detection interrupt

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Select sub clock as subsystem clock

SELLOSC bit ← 0

Control LED lighting (high-speed on-chip oscillator clock)

P5.0 bit ← 1: Set LED3 to OFF
P5.1 bit ← 1: Set LED4 to OFF
P5.2 bit ← 0: Set LED5 to ON

Control LED lighting (CPU operation)

P6.0 bit ← 0: Set LED1 to ON
P6.1 bit ← 0: Set LED2 to ON

Clear external interrupt request request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()

Execute NOP instruction continuously

return

Figure 5.73 Status Transition FtoB(2/2)
5.6.52 Status Transition End Processing

Figure 5.74 shows the flowchart for status transition end processing.

![Flowchart for Status Transition End Processing](image)

- **R_MAIN_END()**
- **Disable interrupt**
  - IE ← 0
- **Control LED lighting (OFF)**
  - P5.0 bit ← 1: Set LED3 to OFF
  - P5.1 bit ← 1: Set LED4 to OFF
  - P5.2 bit ← 1: Set LED5 to OFF
- **Control LED lighting (OFF)**
  - P6.0 bit ← 1: Set LED1 to OFF
  - P6.1 bit ← 1: Set LED2 to OFF
- **return**

**Figure 5.74** Status Transition End Processing
5.6.53 External Interrupt Servicing

Figure 5.75 shows the flowchart for external interrupt servicing.

![Flowchart for External Interrupt Servicing](image)

Figure 5.75 External Interrupt Servicing
5.6.54 12-bit Interval Timer Interrupt Servicing
Figure 5.76 shows the flowchart for 12-bit interval timer interrupt servicing.

```
5.6.55 A/D Conversion Completion Interrupt Servicing
Figure 5.77 shows the flowchart for A/D conversion completion interrupt servicing.
```

```
5.6.55 A/D Conversion Completion Interrupt Servicing
Figure 5.77 shows the flowchart for A/D conversion completion interrupt servicing.
```

```
5.6.55 A/D Conversion Completion Interrupt Servicing
Figure 5.77 shows the flowchart for A/D conversion completion interrupt servicing.
```
6. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

7. Reference Documents

RL78/I1D User’s Manual: Hardware (R01UH0474E)
RL78 Family User’s Manual: Software (R01US0015E)

The latest versions can be downloaded from the Renesas Electronics website.

Technical Update/Technical News

The latest information can be downloaded from the Renesas Electronics website.

Website and Support

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

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

All trademarks and registered trademarks are the property of their respective owners.
<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Jun. 27, 2017</td>
<td>First edition issued</td>
</tr>
</tbody>
</table>
1. 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 LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   - The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   - The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable.
   When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   - When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products
   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.
   - The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.
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 these circuits, software, or information.

2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other disputes 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, drawing, chart, program, algorithm, application examples.

3. No 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 not alter, modify, copy, or otherwise misappropriate 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, copy or otherwise misappropriation of Renesas Electronics products.

5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends 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; and 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.

Renesas Electronics products are neither intended nor 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 damages (space and 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 third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.

6. When using the Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor 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 radiation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions or failure or accident arising out of the use of Renesas Electronics products beyond such specified ranges.

7. 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. Further, Renesas Electronics products are not subject to radiation resistance design. Please ensure to implement safety measures to guard them against the possibility of bodily injury, injury or damage caused by fire, and social damage in the event of 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 by your own responsibility as warranty for your product/system. Because the evaluation of microcomputer software alone is very difficult and not practical, please evaluate the safety of the final products or systems manufactured by you.

8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please investigate applicable laws and regulations that require the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive carefully and sufficiently and use 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.

9. 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 not use Renesas Electronics products or technologies for (1) any purpose relating to the development, design, manufacture, use, stockpiling, etc., of weapons of mass destruction, such as nuclear weapons, chemical weapons, or biological weapons, or missiles (including unmanned aerial vehicles (UAVs)) for delivering such weapons, (2) any purpose relating to the development, design, manufacture, or use of conventional weapons, or (3) any other purpose of disturbing international peace and security, and you shall not sell, export, lease, transfer, or release Renesas Electronics products or technologies to any third party whether directly or indirectly with knowledge or reason to know that the third party or any other party will engage in the activities described above. When exporting, selling, transferring, etc., Renesas Electronics products or technologies, you shall comply with any applicable export control laws and regulations promulgated and administered by the governments of the countries asserting jurisdiction over the parties or transactions.

10. Please acknowledge and agree that you shall bear all the losses and damages which are incurred from the misuse or violation of the terms and conditions described in this document, including this notice, and hold Renesas Electronics harmless, if such misuse or violation results from your resale or making Renesas Electronics products available any third party.

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

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

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics products" means any product developed or manufactured by or for Renesas Electronics.

(Rv.3.0-1 November 2016)