Introduction

This application note describes how to change the RL78/G11’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/G11

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 middle-speed on-chip oscillator clock to high-speed on-chip oscillator clock ... 12
   1.1.5 Changing from middle-speed on-chip oscillator clock to low-speed on-chip oscillator clock ... 13
   1.1.6 Changing from high-speed on-chip oscillator clock to high-speed system clock .............. 14
   1.1.7 Changing from low-speed on-chip oscillator clock to high-speed on-chip oscillator clock ...... 16
   1.1.8 Changing from low-speed on-chip oscillator clock to middle-speed on-chip oscillator clock ... 17
   1.1.9 Changing from low-speed on-chip oscillator clock to high-speed system clock ............... 18
   1.1.10 Changing from high-speed system clock to high-speed on-chip oscillator clock ............... 20
   1.1.11 Changing from high-speed system clock to middle-speed on-chip oscillator clock ....... 21
   1.1.12 Changing from high-speed system clock to low-speed on-chip oscillator clock ............ 22

2. Operation Confirmation Conditions .................................................................................................... 23

3. Related Application Notes .................................................................................................................. 23

4. Hardware Explanation ........................................................................................................................ 24
   4.1 Hardware Configuration Example .................................................................................................. 24
   4.2 Used Pin List ................................................................................................................................ 24

5. Software Explanation ........................................................................................................................ 25
   5.1 Operation Outline ............................................................................................................................ 25
   5.2 Option Byte Settings ...................................................................................................................... 28
   5.3 Variables ....................................................................................................................................... 28
   5.4 Functions (subroutines) .................................................................................................................. 29
   5.5 Function (subroutine) Specifications ............................................................................................... 30
   5.6 Flowcharts .................................................................................................................................... 38
   5.6.1 Initialization Function ................................................................................................................ 38
   5.6.2 System Function .......................................................................................................................... 39
   5.6.3 Input/Output Port Settings ......................................................................................................... 40
   5.6.4 Clock Generator Setting ............................................................................................................ 41
   5.6.5 External Interrupt Setting ......................................................................................................... 46
   5.6.6 12-bit Interval Timer Setting .................................................................................................... 48
   5.6.7 Main Processing ......................................................................................................................... 51
   5.6.8 Status Transition AtoB ............................................................................................................. 56
   5.6.9 CPU operation (NOP instruction execution) ............................................................................. 56
   5.6.10 Status Transition BtoD ............................................................................................................ 57
   5.6.11 Error Processing of Status Transition ................................................................................... 59
   5.6.12 Status Transition DtoL .......................................................................................................... 59
   5.6.13 Status Transition LtoD ........................................................................................................... 60
   5.6.14 Status Transition DtoM .......................................................................................................... 60
   5.6.15 Status Transition MtoD .......................................................................................................... 61
   5.6.16 Status Transition DtoB .......................................................................................................... 62
   5.6.17 Status Transition FtoB .......................................................................................................... 64
   5.6.18 Status Transition FtoB .......................................................................................................... 64
   5.6.19 Status Transition GtoB .......................................................................................................... 65
   5.6.20 Status Transition GtoB .......................................................................................................... 65
   5.6.21 Status Transition BtoH .......................................................................................................... 66
   5.6.22 A/D Converter Setting .......................................................................................................... 67
   5.6.23 A/D Converter Initial Setting ................................................................................................ 68
   5.6.24 Status Transition HtoB .......................................................................................................... 73
   5.6.25 Status Transition BtoC .......................................................................................................... 74
   5.6.26 Status Transition CtoD .......................................................................................................... 76
5.6.27 Status TransitionDtoE ................................................................. 78
5.6.28 Status Transition EtoD ............................................................... 80
5.6.29 Status TransitionDtoC ............................................................... 82
5.6.30 Status Transition CtoI ............................................................... 84
5.6.31 Status Transition ItoC ............................................................... 84
5.6.32 Status Transition CtoJ ............................................................... 85
5.6.33 Status Transition JtoC ............................................................... 85
5.6.34 Status Transition CtoK ............................................................... 86
5.6.35 Status Transition KtoC ............................................................... 87
5.6.36 Status Transition CtoE ............................................................... 88
5.6.37 Status Transition EtoC ............................................................... 90
5.6.38 Status Transition CtoB ............................................................... 92
5.6.39 Status Transition BtoE ............................................................... 94
5.6.40 Status Transition EtoN ............................................................... 96
5.6.41 Status Transition NtoE ............................................................... 96
5.6.42 Status Transition EtoB ............................................................... 97
5.6.43 Status Transition End Processing ............................................. 99
5.6.44 External Interrupt Servicing ................................................. 100
5.6.45 12-bit Interval Timer Interrupt Servicing .................................. 101
5.6.46 A/D Conversion Completion Interrupt Servicing .................... 101

6. Sample Code .................................................................................. 102

7. Reference Documents ..................................................................... 102
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 show 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 P00, P01, P54, P55, P56.</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/ANI0 pin.</td>
</tr>
</tbody>
</table>
Figure 1.1  Operating Mode Status Transition Diagram
<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></td>
<td>LED1</td>
</tr>
<tr>
<td>High-speed on-chip oscillator clock (fIH)</td>
<td>Normal operation mode</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>HALT mode</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>SNOOZE mode</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>STOP mode</td>
<td>OFF</td>
</tr>
<tr>
<td>Middle-speed on-chip oscillator clock (fIM)</td>
<td>Normal operation mode</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>HALT mode</td>
<td>OFF</td>
</tr>
<tr>
<td></td>
<td>SNOOZE mode</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>STOP mode</td>
<td>OFF</td>
</tr>
<tr>
<td>Low-speed on-chip oscillator clock (fIL)</td>
<td>Normal operation mode</td>
<td>ON</td>
</tr>
<tr>
<td></td>
<td>HALT mode</td>
<td>OFF</td>
</tr>
<tr>
<td>High-speed system clock (fMX)</td>
<td>Normal operation mode</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>
</tbody>
</table>
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 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 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
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.

<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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
<td></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>0</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 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.

<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>0</td>
<td>0</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>
<td>1</td>
</tr>
</tbody>
</table>

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.

<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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
<td></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></td>
<td></td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.2 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.

<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>SELLOSC</td>
</tr>
<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: 210\( \mu \)s) using the timer function or another function. In this application note it 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>CLS</td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MCS1</td>
</tr>
<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>

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.

<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>MSTOP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
</tr>
<tr>
<td></td>
<td>x</td>
<td>1</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
1.1.3 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.

<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>EXCLK</td>
<td>OSCSEL</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>AMPH</td>
</tr>
<tr>
<td>0/1</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>

   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.

<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>OSTS</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>OSTS2</td>
<td>OSTS1</td>
</tr>
<tr>
<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>

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>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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
</tr>
<tr>
<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>

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.

<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>MOST8</td>
<td>MOST9</td>
<td>MOST10</td>
<td>MOST11</td>
<td>MOST13</td>
<td>MOST15</td>
<td>MOST17</td>
<td>MOST18</td>
</tr>
<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>
</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>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>0</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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.4 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 fCLK 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>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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
<td></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>0</td>
<td></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>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>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></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>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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></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 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 \( f_{\text{CLK}} \) 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 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>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>1</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>MSTOP</th>
<th>1</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>1</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
1.1.6 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>
<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>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>AMPH</td>
</tr>
<tr>
<td>0/1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</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 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>OSTS0</td>
<td>OSTS1</td>
<td>OSTS2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>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>
<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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
</tr>
<tr>
<td>0</td>
<td>1</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>

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.

<table>
<thead>
<tr>
<th>OSTC</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>MOST8</td>
<td>MOST9</td>
<td>MOST10</td>
<td>MOST11</td>
<td>MOST13</td>
<td>MOST15</td>
<td>MOST17</td>
<td>MOST18</td>
<td></td>
</tr>
<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></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>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>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>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>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</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
1.1.7 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 \( f_{\text{CLK}} \) 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 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.

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

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.

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

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.

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

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
1.1.8 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 $f_{CLK}$ 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. 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>1</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>

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, 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.

<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>SELLOSC</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
1.1.9 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 \( f_x > 10\text{MHz} \), 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>
<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>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>AMPH</td>
</tr>
<tr>
<td>OSCSEL</td>
<td>0/1</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>

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 \( \mu \text{s} \) 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>OSTS2</td>
<td>OSTS1</td>
<td>OSTS0</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>1</td>
<td>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>
<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>1</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>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td></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 \( \mu \text{s} \) based on a 10 MHz resonator.

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

<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>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSS</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCS</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCM0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCM1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCM2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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.

<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>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SELLOSC</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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 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.

\[
\begin{array}{cccccccc}
\text{CSC} & \text{MSTOP} & 1 & 0 & 0 & 0 & 0 & \text{MIOEN} & \text{HIOSTOP} \\
0 & 1 & 0 & 0 & 0 & 0 & x & 0 \\
\end{array}
\]

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: 65\(\mu\)s) 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.

\[
\begin{array}{cccccccc}
\text{CKC} & \text{CLS} & \text{CSS} & \text{MCS} & \text{MCM0} & 0 & 0 & \text{MCS1} & \text{MCM1} \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

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.

\[
\begin{array}{cccccccc}
\text{CSC} & \text{MSTOP} & 1 & 0 & 0 & 0 & 0 & \text{MIOEN} & \text{HIOSTOP} \\
1 & 1 & 0 & 0 & 0 & 0 & x & 0 \\
\end{array}
\]

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.11 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 fCLK 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>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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</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>x</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>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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</td>
</tr>
<tr>
<td>0</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>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></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>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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MIOEN</td>
<td>HIOSTOP</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Register setting values:
- x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit
1.1.12 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 $f_{CLK}$ 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.

<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>SELLOSC</td>
</tr>
<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>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>1</td>
<td>1</td>
<td>1</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 1, set (1) the MSTOP bit and stop the oscillating the X1 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>1</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>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

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>
<thead>
<tr>
<th>Item</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCU used</td>
<td>RL78/G11 (R5F1056A)</td>
</tr>
<tr>
<td>Operating frequencies</td>
<td>- High-speed on-chip oscillator clock: 24MHz</td>
</tr>
<tr>
<td></td>
<td>- Middle-speed on-chip oscillator clock: 4MHz</td>
</tr>
<tr>
<td></td>
<td>- Low-speed on-chip oscillator clock: 15kHz</td>
</tr>
<tr>
<td></td>
<td>- High-speed system clock: 20MHz</td>
</tr>
<tr>
<td></td>
<td>- CPU/peripheral hardware clock: 24MHz/20MHz/4MHz/15kHz&lt;sup&gt;Note&lt;/sup&gt;</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>3.3V (operating range 2.9V to 5.5V)</td>
</tr>
<tr>
<td></td>
<td>LVD operations ($V_{LVD}$): reset mode 2.81V (2.76V to 2.87V)</td>
</tr>
<tr>
<td>Integrated development</td>
<td>CS+ for CC V4.01.00 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>environment (CS+)</td>
<td></td>
</tr>
<tr>
<td>C compiler (CS+)</td>
<td>CC-RL V1.02.00 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>Integrated development</td>
<td>e² studio V5.2.0.020 from Renesas Electronics Corp.</td>
</tr>
<tr>
<td>environment (e² studio)</td>
<td></td>
</tr>
<tr>
<td>C compiler (e² studio)</td>
<td>CC-RL V1.02.00 from Renesas Electronics Corp.</td>
</tr>
</tbody>
</table>

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 CC-RL (R01AN3095E) 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.

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 (VLVD) set in LVD.

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/AIN0</td>
<td>Input</td>
<td>A/D converter analog input port</td>
</tr>
<tr>
<td>P00</td>
<td>Output</td>
<td>LED (LED1) control port</td>
</tr>
<tr>
<td>P01</td>
<td>Output</td>
<td>LED (LED2) control port</td>
</tr>
<tr>
<td>P54</td>
<td>Output</td>
<td>LED (LED3) control port</td>
</tr>
<tr>
<td>P55</td>
<td>Output</td>
<td>LED (LED4) control port</td>
</tr>
<tr>
<td>P56</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 31 of Figure 1.1 operation mode status transition diagram.

(1) Input/output port initialization

- P00-P01 and P54-P56 pins: set as output ports (use to control LEDs)
- P137/INTP0 pin: set as input port (use for switch input)
- P20/ANI0 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 (000C2H/010C2H.)
- High-speed on-chip oscillator clock frequency: set to 24 MHz
- 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 \( f_{\text{MAIN}} \) as the CPU/peripheral hardware clock \( f_{\text{CLK}} \).

(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) 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>(3) High-speed system clock</td>
<td>HALT mode</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(4) 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>(5) High-speed system clock</td>
<td>STOP mode</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</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) 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>(8) 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>(9) 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>(10) 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>(11) 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>(12) 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>(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) 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>(15) 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>(16) 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>OFF</td>
</tr>
<tr>
<td>(17) 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>(18) 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>(19) Middle-speed on-chip</td>
<td>HALT mode</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>oscillator clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(20) Middle-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(21) Middle-speed on-chip</td>
<td>STOP mode</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>oscillator clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(22) Middle-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(23) Middle-speed on-chip</td>
<td>SNOOZE mode</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>oscillator clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(24) Middle-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(25) Low-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(26) Middle-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(27) High-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(28) Low-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(29) Low-speed on-chip</td>
<td>HALT mode</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>oscillator clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(30) Low-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(31) High-speed on-chip</td>
<td>Normal operation</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>oscillator clock</td>
<td>mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After the CPU clock and operation mode have been changed according to steps 1 to 31 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/G11 User’s Manual for usage notes concerning this device.
5.2 Option Byte Settings

Table 5.1 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, HOCO: 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 5.2 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_BtoD, R_MAIN_DtoB, R_MAIN_BtoC, R_MAIN_CtoD, R_MAIN_DtoE, R_MAIN_EtoD, R_MAIN_DtoC, R_MAIN_CtoE, R_MAIN_EtoC, R_MAIN_CtoB, R_MAIN_BtoE, R_MAIN_EtoB, 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 5.3 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_BtoD</td>
<td>Status transition processing from (B) to (D)</td>
<td>(2)</td>
</tr>
<tr>
<td>R_MAIN_DtoL</td>
<td>Status transition processing from (D) to (L)</td>
<td>(3)</td>
</tr>
<tr>
<td>R_MAIN_LtoD</td>
<td>Status transition processing from (L) to (D)</td>
<td>(4)</td>
</tr>
<tr>
<td>R_MAIN_DtoM</td>
<td>Status transition processing from (D) to (M)</td>
<td>(5)</td>
</tr>
<tr>
<td>R_MAIN_MtoC</td>
<td>Status transition processing from (M) to (D)</td>
<td>(6)</td>
</tr>
<tr>
<td>R_MAIN_DtoB</td>
<td>Status transition processing from (D) to (B)</td>
<td>(7)</td>
</tr>
<tr>
<td>R_MAIN_BtoF</td>
<td>Status transition processing from (B) to (F)</td>
<td>(8)</td>
</tr>
<tr>
<td>R_MAIN_BtoG</td>
<td>Status transition processing from (B) to (G)</td>
<td>(9)</td>
</tr>
<tr>
<td>R_MAIN_GtoB</td>
<td>Status transition processing from (G) to (B)</td>
<td>(10)</td>
</tr>
<tr>
<td>R_MAIN_BtoH</td>
<td>Status transition processing from (B) to (H)</td>
<td>(11)</td>
</tr>
<tr>
<td>R_MAIN_HtoB</td>
<td>Status transition processing from (H) to (B)</td>
<td>(12)</td>
</tr>
<tr>
<td>R_MAIN_BtoC</td>
<td>Status transition processing from (B) to (C)</td>
<td>(13)</td>
</tr>
<tr>
<td>R_MAIN_CtoD</td>
<td>Status transition processing from (C) to (D)</td>
<td>(14)</td>
</tr>
<tr>
<td>R_MAIN_DtoE</td>
<td>Status transition processing from (D) to (E)</td>
<td>(15)</td>
</tr>
<tr>
<td>R_MAIN_EtoD</td>
<td>Status transition processing from (E) to (D)</td>
<td>(16)</td>
</tr>
<tr>
<td>R_MAIN_DtoC</td>
<td>Status transition processing from (D) to (C)</td>
<td>(17)</td>
</tr>
<tr>
<td>R_MAIN_CtoI</td>
<td>Status transition processing from (C) to (I)</td>
<td>(18)</td>
</tr>
<tr>
<td>R_MAIN_ItoC</td>
<td>Status transition processing from (I) to (C)</td>
<td>(19)</td>
</tr>
<tr>
<td>R_MAIN_CtoJ</td>
<td>Status transition processing from (C) to (J)</td>
<td>(20)</td>
</tr>
<tr>
<td>R_MAIN_JtoC</td>
<td>Status transition processing from (J) to (C)</td>
<td>(21)</td>
</tr>
<tr>
<td>R_MAIN_CtoK</td>
<td>Status transition processing from (C) to (K)</td>
<td>(22)</td>
</tr>
<tr>
<td>R_MAIN_KtoC</td>
<td>Status transition processing from (K) to (C)</td>
<td>(23)</td>
</tr>
<tr>
<td>R_MAIN_CtoE</td>
<td>Status transition processing from (C) to (E)</td>
<td>(24)</td>
</tr>
<tr>
<td>R_MAIN_EtoC</td>
<td>Status transition processing from (E) to (C)</td>
<td>(25)</td>
</tr>
<tr>
<td>R_MAIN_CtoB</td>
<td>Status transition processing from (C) to (B)</td>
<td>(26)</td>
</tr>
<tr>
<td>R_MAIN_BtoE</td>
<td>Status transition processing from (B) to (E)</td>
<td>(27)</td>
</tr>
<tr>
<td>R_MAIN_EtoN</td>
<td>Status transition processing from (E) to (N)</td>
<td>(28)</td>
</tr>
<tr>
<td>R_MAIN_NtoE</td>
<td>Status transition processing from (N) to (E)</td>
<td>(29)</td>
</tr>
<tr>
<td>R_MAIN_EtoB</td>
<td>Status transition processing from (E) to (B)</td>
<td>(30)</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.

[Function Name] R_MAIN_AtoB

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

[Function Name] R_MAIN_BtoD

Outline  Status transition processing from (B) to (D)
Declaration void R_MAIN_BtoD(void)
Description Change the CPU clock from high-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_DtoL

Outline  Status transition processing from (D) to (L)
Declaration void R_MAIN_DtoL(void)
Description Control LED lighting, then transition to HALT mode.
              (CPU clock stopped (when using high-speed system clock))
Argument  None
Return Value None
Notes     None

[Function Name] R_MAIN_LtoD

Outline  Status transition processing from (L) to (D)
Declaration void R_MAIN_LtoD(void)
Description Control LED lighting.
              (CPU clock: high-speed system clock)
Argument  None
Return Value None
Notes     None

[Function Name] R_MAIN_DtoM

Outline  Status transition processing from (D) to (M)
Declaration void R_MAIN_DtoM(void)
Description Control LED lighting, then transition to STOP mode.
              (Stop CPU clock (when using high-speed system clock))
Argument  None
Return Value None
Notes     None
[Function Name] 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.</td>
</tr>
<tr>
<td></td>
<td>(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_BtoF

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (B) to (F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_BtoF(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting, then transition to HALT mode.</td>
</tr>
<tr>
<td></td>
<td>(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_FtoB

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (F) to (B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_FtoB(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting.</td>
</tr>
<tr>
<td></td>
<td>(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

<table>
<thead>
<tr>
<th>Outline</th>
<th>Status transition processing from (B) to (G)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>void R_MAIN_BtoG(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Control LED lighting, then transition to STOP mode.</td>
</tr>
<tr>
<td></td>
<td>(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_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_DtoE

**Outline**
Status transition processing from (D) to (E)

**Declaration**
void R_MAIN_DtoE(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_EtoD

**Outline**
Status transition processing from (E) to (D)

**Declaration**
void R_MAIN_EtoD(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_CtoI

**Outline**
Status transition processing from (C) to (I)

**Declaration**
void R_MAIN_CtoI(void)

**Description**
Control LED lighting, then transition to HALT mode. (CPU clock stopped (when using middle-speed on-chip oscillator clock))

**Argument**
None

**Return Value**
None

**Notes**
None

[Function Name] R_MAIN_ItoC

**Outline**
Status transition processing from (I) to (C)

**Declaration**
void R_MAIN_ItoC(void)

**Description**
Control LED lighting. (CPU clock: middle-speed on-chip oscillator clock)

**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 STOP mode.
| (CPU clock stopped (when using middle-speed on-chip oscillator clock))
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 | Set A/D converter and control LED lighting.
| Then, transition to SNOOZE mode.
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 | 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 low-speed
| on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument | None
Return Value | None
Notes | None
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Outline</th>
<th>Declaration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_MAIN_EtoC</td>
<td>Status transition processing from (E) to (C)</td>
<td>void R_MAIN_EtoC(void)</td>
<td>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.</td>
</tr>
<tr>
<td>R_MAIN_CtoB</td>
<td>Status transition processing from (C) to (B)</td>
<td>void R_MAIN_CtoB(void)</td>
<td>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.</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 low-speed on-chip oscillator clock. After the clock is switched, control LED lighting.</td>
</tr>
<tr>
<td>R_MAIN_EtoN</td>
<td>Status transition processing from (E) to (N)</td>
<td>void R_MAIN_EtoN(void)</td>
<td>Control LED lighting, then transition to HALT mode. (CPU clock stopped (when using low-speed on-chip oscillator clock))</td>
</tr>
<tr>
<td>R_MAIN_NtoE</td>
<td>Status transition processing from (N) to (E)</td>
<td>void R_MAIN_NtoE(void)</td>
<td>Control LED lighting. (CPU clock: low-speed on-chip oscillator clock)</td>
</tr>
<tr>
<td>Function Name</td>
<td>Outline</td>
<td>Declaration</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------</td>
<td>--------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</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 low-speed on-chip oscillator clock to high-speed on-chip oscillator clock. After the clock is switched, control LED lighting.</td>
</tr>
<tr>
<td>R_MAIN_NOP_Loop</td>
<td>Continuous NOP instruction execution processing</td>
<td>void R_MAIN_NOP_Loop(void)</td>
<td>Execute NOP instruction continuously. End processing when external interrupt generation detection flag is confirmed.</td>
</tr>
<tr>
<td>R_MAIN_END</td>
<td>End processing of status transition</td>
<td>void R_MAIN_END(void)</td>
<td>Disable interrupts. Control LED lighting (all off).</td>
</tr>
<tr>
<td>R_MAIN_ERROR</td>
<td>Error processing of status transition</td>
<td>void R_MAIN_ERROR(void)</td>
<td>Disable interrupts. Control LED lighting (all off). Loop processing with in the function(Return is only reset input).</td>
</tr>
<tr>
<td>R_MAIN_AD_SnoozeOn</td>
<td>A/D converter setting</td>
<td>void R_MAIN_AD_SnoozeOn(void)</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Function Name: r_intc0_interrupt

<table>
<thead>
<tr>
<th>Outline</th>
<th>External interrupt generation detection flag confirmation processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>static void __near r_intc0_interrupt(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Confirm external interrupt generation detection flag with generation of external interrupt. End processing when switch input level changes to high.</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_it_interrupt

<table>
<thead>
<tr>
<th>Outline</th>
<th>12-bit interval timer interval signal detection interrupt count processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>static void __near r_it_interrupt(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Increment the RITCOUNT each time the 12-bit interval timer interrupt signal detection interrupt is generated.</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_adc_interrupt

<table>
<thead>
<tr>
<th>Outline</th>
<th>SNOOZE mode release processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>static void __near r_adc_interrupt(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Clear the AWC bit of the ADM2 register and release the SNOOZE mode.</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>
5.6 Flowcharts

Figure 5.1 shows the entire flow for this application note.

![Overall Flowchart](image1)

Reference the option byte before executing the CPU initialization function.

5.6.1 Initialization Function

Figure 5.2 shows the flowchart for the initialization function.

![Initialization Function](image2)
5.6.2 System Function

Figure 5.3 shows the flowchart for system function.

```
R_Systeminit()

Disuse peripheral I/O Redirection function

Set up I/O ports R_PORT_Create()

Set up CPU clock R_CGC_Create()

Set interrupt R_IT_Create()

Set interrupt R_INTC_Create()

Disabled the invalid memory access detection

IAWCTL register ← 00H

PIOR0 register ← 00H
PIOR1 register ← 00H
PIOR2 register ← 00H
PIOR3 register ← 00H

return
```

Figure 5.3 System Function
5.6.3 Input/Output Port Settings

Figure 5.4 shows the flowchart for the input/output port settings.

```
R_PORT_Create()

Set analog input Alternate-function pins

Set port mode registers and port registers

return

PM0 ← 11111100B: Set P0.1 to P0.0 to input ports
PM2 ← 01110001B: Set P2.0 to input ports
PM4 ← 11111111B: Set P4.0 to input ports
PM5 ← 10001111B: Set P5.6 to P5.4 to out ports

PO ← 00000011B: Set P0.1 to P0.0 to high-level output
P5 ← 01110000B: Set P5.6 to P5.4 to high-level output

PM0 register ← 1111100B: Set P0.1 to P0.0 to digital input/output
PM2 register ← 11110001B: Set P2.0 to analog input
PM5 register ← 10111111B: Set P5.6 to digital input/output
```

**Figure 5.4 Input/Output Port Settings**

Note: Refer to the initialization flowchart in the RL78/G13 Initialization (R01AN2575J) 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 \( V_{DD} \) or \( V_{SS} \) through a resister.
5.6.4 Clock Generator Setting

Figure 5.5 shows the flowchart for setting the clock generator.

```
R_CGC_Create()

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

Set high-speed system clock/subsystem clock
  CMC register ← 01010101B: X1 oscillation mode
  X1 clock: 10 MHz ≤ fX ≤ 20 MHz
  OSTC register ← 00000111B

Set clock operation controls
  MSTOP bit ← 1: X1 oscillator stopped

Set low-speed on-chip oscillator clock
  SELLOSC bit ← 1: Select low-speed on-chip oscillator clock
  as Subsystem clock

Oscillation stabilization wait time

Select CPU/peripheral hardware clock (fCLK)
  OSMC register ← 10H: Provide fIL to 12-bit interval timer
  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 oscillator clock (fOCO)

return
```

Figure 5.5 Clock Generator Setting
Clock operation mode setting

- Clock operation mode control register (CMC)
  - High-speed system clock pin operation mode: X1 oscillation mode
  - X1 clock oscillation frequency control: $10\text{MHz} < f_X \leq 20\text{MHz}$

Symbol: CMC

<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>EXCLK</td>
<td>OSCSEL</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>AMPH</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Bits 7-6

<table>
<thead>
<tr>
<th>EXCLK</th>
<th>OSCSEL</th>
<th>High-seed oscillation clock pin operation mode</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>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>X1 oscillation mode</td>
<td>Crystal/ceramic resonator connection</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>input port mode</td>
<td>input port</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>External clock input mode</td>
<td>input port</td>
<td>External clock input</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>$1\text{MHz} \leq f_X \leq 10\text{MHz}$</td>
</tr>
<tr>
<td>1</td>
<td>$10\text{MHz} &lt; f_X \leq 20\text{MHz}$</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/G11 User’s Manual (Hardware version) 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
  - 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>MSTOP</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>MIOEN</th>
<th>HIOSTOP</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSTOP</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>
<td>0</td>
</tr>
</tbody>
</table>

Bit 7

| MSTOP | High-speed system clock operation control
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X1 oscillation mode</td>
</tr>
<tr>
<td>0</td>
<td>X1 oscillator operating</td>
</tr>
<tr>
<td>1</td>
<td>X1 oscillator stopped</td>
</tr>
</tbody>
</table>

Bit 1

<table>
<thead>
<tr>
<th>MIOEN</th>
<th>Middle-speed on-chip oscillator clock operation control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Middle-speed on-chip oscillator stopped</td>
</tr>
<tr>
<td>1</td>
<td>Middle-speed on-chip oscillator operating</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>HIOSTOP</th>
<th>High-speed on-chip oscillator clock operation control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>High-speed on-chip oscillator operating</td>
</tr>
<tr>
<td>1</td>
<td>High-speed on-chip oscillator stopped</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.
CPU/peripheral hardware clock (fCLK) setting

- System clock control register (CKC)
  Status of fCLK: main system clock
  Selection of fCLK: high-speed on-chip oscillator clock (fIH)

Symbol: CKC

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLS</td>
<td>CSS</td>
<td>MCS</td>
<td>MCM0</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>
</tr>
</tbody>
</table>

Bit 7

<table>
<thead>
<tr>
<th>CLS</th>
<th>Status of CPU/peripheral hardware clock (fCLK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Main system clock (fMAIN)</td>
</tr>
<tr>
<td>1</td>
<td>Subsystem clock (fSUB)</td>
</tr>
</tbody>
</table>

Bit 6

<table>
<thead>
<tr>
<th>CSS</th>
<th>Selection of CPU/peripheral hardware clock (fCLK)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Main system clock (fMAIN)</td>
</tr>
<tr>
<td>1</td>
<td>Subsystem clock (fSUB)</td>
</tr>
</tbody>
</table>

Bit 5

<table>
<thead>
<tr>
<th>MCS</th>
<th>Status of main system clock (fMAIN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>High-speed on-chip oscillator clock (fIH)</td>
</tr>
<tr>
<td>1</td>
<td>High-speed system clock (fMX)</td>
</tr>
</tbody>
</table>

Bit 4

<table>
<thead>
<tr>
<th>MCM0</th>
<th>Main system clock (fMAIN) operation control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Selects high-speed on-chip oscillator clock (fIH) as main system clock (fMAIN)</td>
</tr>
<tr>
<td>1</td>
<td>Selects high-speed system clock (fMX) as main system clock (fMAIN)</td>
</tr>
</tbody>
</table>

Bit 1

<table>
<thead>
<tr>
<th>MCS1</th>
<th>Status of main on-chip oscillator clock (foco)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>High-speed on-chip oscillator clock (fIH)</td>
</tr>
<tr>
<td>1</td>
<td>Middle-speed on-chip oscillator clock (fIM)</td>
</tr>
</tbody>
</table>

Bit 0

<table>
<thead>
<tr>
<th>MCS1</th>
<th>Main on-chip oscillator clock (foco) operation control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>High-speed on-chip oscillator clock (fIH)</td>
</tr>
<tr>
<td>1</td>
<td>Middle-speed on-chip oscillator clock (fIM)</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.
Operation speed mode control

- Operation speed mode control register (OSMC)
  Selection of operation clock for 12-bit interval timer: low-speed on-chip oscillator clock

Symbol: OSMC

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

Bit 4

<table>
<thead>
<tr>
<th>WUTMMCK0</th>
<th>Selection of operation clock for 12-bit interval timer</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Do not select low-speed on-chip oscillator clock</td>
</tr>
<tr>
<td>1</td>
<td>Select low-speed on-chip oscillator clock</td>
</tr>
</tbody>
</table>

Subsystem clock select

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

Symbol: CKSEL

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

Bit 0

<table>
<thead>
<tr>
<th>SELLOSC</th>
<th>Selection of low-speed on-chip oscillator clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Do not select low-speed on-chip oscillator clock</td>
</tr>
<tr>
<td>1</td>
<td>Select low-speed on-chip oscillator clock</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.
5.6.5 External Interrupt Setting

Figure 5.6 shows the flowchart for setting the external interrupt.

![Flowchart for setting the external interrupt](image)

- **R_INTC_Create()**
  - **Disable all INTP interrupta**
    - MK0L register
      - PMK5-PMK0 bit ← 111111B : Interrupt processing disabled
    - MK1H register
      - PMK11-PMK9 bit ← 111B : Interrupt processing disabled
    - MK2H register
      - FOMK bit ← 1 : Interrupt processing disabled
    - IF0L register
      - PIF5-PIF0 bit ← 1111111B : Interrupt request signal not generated
    - IF1H register
      - PIF11-PIF9 bit ← 111B : Interrupt request signal not generated
    - IF2H register
      - FOIF 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**
    - EGP1 bit ← 0
    - EGN1 bit ← 1: Specify falling edge
      as INTP0 pin valid edge

**Figure 5.6 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>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>EGP7</td>
<td>EGP6</td>
<td>EGP5</td>
<td>EGP4</td>
<td>EGP3</td>
<td>EGP2</td>
<td>EGP1</td>
<td>EGP0</td>
</tr>
<tr>
<td>x</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>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>EGN7</td>
<td>EGN6</td>
<td>EGN5</td>
<td>EGN4</td>
<td>EGN3</td>
<td>EGN2</td>
<td>EGN1</td>
<td>EGN0</td>
</tr>
<tr>
<td>x</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>EGP1</th>
<th>EGN1</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/G11 User’s Manual (Hardware version) for details on how to set registers.
5.6.6 12-bit Interval Timer Setting

Figure 5.7 shows the flowchart for setting the 12-bit interval timer.

![Flowchart for 12-bit Interval Timer Setting](image)

- **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
  - Select low-speed on-chip oscillator clock for 12-bit interval timer clock
    - TPS3 register ← 00H
  - Set the compare value (5ms) of 12-bit interval timer
    - ITMC register ← 004AH
  - return

**Figure 5.7** 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>Bit 7</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMKAEN</td>
<td>Control of 12-bit interval timer input clock supply</td>
</tr>
<tr>
<td>0</td>
<td>Stops input clock supply.</td>
</tr>
<tr>
<td>1</td>
<td>Enables input clock supply.</td>
</tr>
</tbody>
</table>

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

Symbol: MK1H

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

Note: Refer to the RL78/G11 User’s Manual (Hardware version) 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

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>RINTE</th>
<th>12-bit interval timer operation control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Count operation stopped (count clear)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Count operation started</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bits 11-0</th>
<th>ITCMP11-ITCMP0</th>
<th>Specification of 12-bit interval timer compare value</th>
</tr>
</thead>
<tbody>
<tr>
<td>000H</td>
<td>Setting prohibited</td>
<td></td>
</tr>
</tbody>
</table>

04AH
These bits generate an interrupt at the fixed cycle (count clock cycles x (ITCMP setting value 04AH + 1)).

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.
5.6.7 Main Processing

Figure 5.8, Figure 5.9, Figure 5.10, Figure 5.11 shows the flowchart for the main processing.

```
main()

Enable interrupt

Enable external interrupt

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

Status transition (B)→(D)
R_MAIN_BtoD()

Status transition (D)→(L)
R_MAIN_DtoL()

Status transition (L)→(D)
R_MAIN_LtoD()

Status transition (D)→(M)
R_MAIN_DtoM()

Status transition (M)→(D)
R_MAIN_MtoD()

Status transition (D)→(B)
R_MAIN_DtoB()

Status transition (B)→(F)
R_MAIN_BtoF()

IE ← 1

PIF0 bit ← 0
PMK0 bit ← 0

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 high-speed system clock operation (D).

Transition to HALT mode (L) while CPU is operating with high-speed system clock (D).

Release HALT mode (L), then transition to high-speed system clock operation (D).

Transition to STOP mode (M) while CPU is operating with high-speed system clock (D).

Release STOP mode (M), then transition to high-speed system clock operation (D).

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

Transition to HALT mode (F) while CPU is operating with high-speed on-chip oscillator clock (B).
```

Figure 5.8 Main Processing (1/4)
Release HALT mode (F), then transition to high-speed on-chip oscillator clock operation (B)

Transition to STOP mod (G) while CPU is operating with high-speed on-chip oscillator clock (B)

Release STOP mode (G), then transition to high-speed on-chip oscillator operation (B)

Transition to SNOOZE mode (H) while CPU is operating with high-speed on-chip oscillator clock (B)

Release SNOOZE mode (H), then transition to high-speed on-chip oscillator operation (B)

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

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

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

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

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

Transition to HALT mode (I) while CPU is operating with middle-speed on-chip oscillator clock (C)

**Figure 5.9** Main Processing (2/4)
Figure 5.10  Main Processing (3/4)
Set LED to OFF
End processing of status transition
R_MAIN_END()
return

Status transition (N)→(E)
R_MAIN_NtoE()
Release HALT mode (N), then transition to middle-speed on-chip oscillator clock operation (E)

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

Figure 5.11 Main Processing (4/4)
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

<table>
<thead>
<tr>
<th></th>
<th>PIF5</th>
<th>PIF4</th>
<th>PIF3</th>
<th>PIF2</th>
<th>PIF1</th>
<th>PIF0</th>
<th>LVIF</th>
<th>WDTIF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</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

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

Symbol: MK0L

<table>
<thead>
<tr>
<th></th>
<th>PMK5</th>
<th>PMK4</th>
<th>PMK3</th>
<th>PMK2</th>
<th>PMK1</th>
<th>PMK0</th>
<th>LVIMK</th>
<th>WDTMK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</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

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

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.
5.6.8 Status Transition AtoB

Figure 5.12 shows the flowchart for status transition AtoB.

![Flowchart for status transition AtoB](image)

- **R_MAIN_AtoB()**
  - Control LED lighting (high-speed on-chip oscillator clock)
    - P5.6 bit ← 0, P5.5, P5.4 bits ← 1: Set LED5 to ON and LED3, LED4 to OFF
  - Control LED lighting (CPU operation)
    - P0.1, P0.0 bits ← 0: Set LED1, 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 5.12 Status Transition AtoB**

5.6.9 CPU operation (NOP instruction execution)

Figure 5.13 shows the flowchart for the CPU operation (NOP instruction execution)

![Flowchart for CPU operation (NOP instruction execution)](image)

- **R_MAIN_NOP_Loop()**
  - Clear external interrupt generation detection flag
  - Execute NOP instruction (512 times)
    - Is external interrupt generation detection flag 1?
      - NO
      - YES (branch when g_int_flg=1)
  - return

**Figure 5.13 CPU Operation (NOP instruction execution)**
5.6.10 Status Transition BtoD

Figure 5.14 and Figure 5.15 shows the flowchart for status transition BtoD.

![Flowchart for status transition BtoD](image)

**Figure 5.14 Status Transition BtoD (1/2)**
Figure 5.15  Status Transition BtoD (2/2)
5.6.11 Error Processing of Status Transition
Figure 5.16 shows the flowchart for error processing of status transition.

![Figure 5.16 Error Processing of Status Transition](image)

5.6.12 Status Transition DtoL
Figure 5.17 shows the flowchart for status transition DtoL.

![Figure 5.17 Status Transition DtoL](image)
5.6.13 Status Transition LtoD
Figure 5.18 shows the flowchart for status transition LtoD.

```
R_MAIN_LtoD()

Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1 and 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 5.18 Status Transition LtoD

5.6.14 Status Transition DtoM
Figure 5.19 shows the flowchart for status transition DtoM.

```
R_MAIN_DtoM()

Control LED lighting (STOP mode)
P0.1, P0.0 bits ← 1: Set LED1, LED2 to OFF

Clear external interrupt request flag
PIF0 bit ← 0

STOP mode

return
```

Figure 5.19 Status Transition DtoM
5.6.15 Status Transition MtoD

Figure 5.20 shows the flowchart for status transition MtoD.

```
R_MAIN_MtoD()

Control LED lighting (STOP mode)

Clear external interrupt request flag

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON

PIF0 bit ← 0

Execute NOP instruction continuously.

return
```

**Figure 5.20** Status Transition MtoD
5.6.16 Status Transition DtoB

Figure 5.21 and Figure 5.22 shows the flowchart for status transition DtoB.

**Figure 5.21 Status Transition DtoB (1/2)**
Figure 5.22  Status Transition DtoB(2/2)

A

Start 12-bit interval timer count

ITMC register ← 8001H

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

No

Disable interval detection interrupt

TMKAMK bit ← 1

Stop 12-bit interval timer count

ITMC register ← 0000H

Stop X1 oscillator

MSTOP bit ← 1

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

P5.6 bit ← 0, P5.5, P5.4 bits ← 1: Set LED3, LED4 to OFF and LED5 to ON

P5.6 bit ← 0, P5.5, P5.4 bits ← 1: Set LED3, LED4 to OFF and LED5 to ON

Control LED lighting (CPU operation)

P0.1, P0.0 bits ← 0: Set LED1, 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
5.6.17 Status Transition BtoF

Figure 5.23 shows the flowchart for status transition BtoF.

**Figure 5.23 Status Transition BtoF**

- **R_MAIN_BtoF()**
  - Control LED lighting (HALT mode)
    - P0.1 bit ← 0, P0.0 bit ← 1: Set LED1 to OFF and LED2 to ON
  - Clear external interrupt request flag
    - PIF0 bit ← 0
  - HALT
  - return

5.6.18 Status Transition FtoB

Figure 5.24 shows the flowchart for status transition FtoB.

**Figure 5.24 Status Transition FtoB**

- **R_MAIN_FtoB()**
  - Control LED lighting (CPU operation)
    - P0.1, P0.0 bits ← 0: Set LED1, 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
5.6.19 Status Transition BtoG

Figure 5.25 shows the flowchart for status transition BtoG.

![Figure 5.25 Status Transition BtoG]

R_MAIN_BtoG()

Control LED lighting (STOP mode)

Clear external interrupt request flag

STOP mode

return

5.6.20 Status Transition GtoB

Figure 5.26 shows the flowchart for status transition GtoB.

![Figure 5.26 Status Transition GtoB]

R_MAIN_GtoB()

Control LED lighting (CPU operation)

Clear external interrupt request flag

CPU operation (NOP instruction execution)

R_MAIN_NOP_Loop()

return

P0.1, P0.0 bits ← 1: Set LED1, LED2 to OFF

PIF0 bit ← 0

P0.1, P0.0 bits ← 0; Set LED1, LED2 to ON

PIF0 bit ← 0

Execute NOP instruction continuously.
5.6.21 Status Transition BtoH

Figure 5.27 shows the flowchart for status transition BtoH.

```
R_MAIN_BtoH()
  
  Disable external interrupt
  PMK0 bit ← 1

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

  R_MAIN_AD_SnoozeOn()

  Disable interval detection
  interrupt
  TMKAMK bit ← 1

  Control LED lighting
  (SNOOZE mode)
  P0.1 bit ← 1, P0.0 bit ← 0: set LED1 to ON and LED2 to OFF

  Start 12-bit interval timer count

  STOP

  ITMC register ← 8FFFH

  return
```

Figure 5.27 Status Transition BtoH
5.6.22 A/D Converter Setting

Figure 5.28 shows the flowchart for setting the A/D converter.

```
R_MAIN_AD_SnoozeOn()

Set A/D converter
R_ADC_Create()

Clear A/D conversion
complete interrupt request flag
ADIF bit ← 0

Enable A/D conversion
interrupt
ADMK bit ← 0

Use SNOOZE mode function
AWC bit ← 1

return
```

**Figure 5.28 A/D Converter Setting**

**SNOOZE mode setting**

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

Symbol: ADM2

<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>ADREFP</td>
<td>ADREFP</td>
<td>ADREFM</td>
<td>0</td>
<td>ADCRK</td>
<td>AWC</td>
<td>0</td>
<td>ADTYP</td>
</tr>
<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/G11 User’s Manual (hardware version) for detailed explanations on how to set registers.
5.6.23 A/D Converter Initial Setting

Figure 5.29 shows the flowchart for setting the A/D converter.

```plaintext
R_MAIN_ADC_Create()

Reset A/D converter

Supply clock to A/D converter

Set A/D conversion to invalid

Clear A/D conversion complete interrupt request flag

Enable A/D conversion interrupt

Set interrupt priority level

Set P20 pin to analog input

A/D converter initial setting:
- Conversion time: 36us
- Select mode
- Hardware trigger mode
- One-shot conversion mode

Reference voltage source setting:
- 10-bit resolution
- Set VDD as + side reference voltage source
- Set VSS as - side reference voltage source

Conversion result comparison upper/lower limit settings:
- Upper limit value: FFH
- Lower limit value: 80H

Analog input channel specification:
- Set AIN0 as analog input channel

ADRES bit ← 1
ADRES bit ← 0
ADCE bit ← 1
ADM0 register ← 00H
ADIF bit ← 0
ADMK bit ← 0
ADPR1 bit ← 1
ADPR0 bit ← 1
PM2.0 bit ← 1
PM2.0 bit ← 1
ADM0 register:
FR0-FR3 bits ← 001B; fCLK/32(fCLK=24MHz)
LV1-LV0 bits ← 00B; Normal mode
ADMD bit ← 0; Select mode
ADCE bit ← 1; Enables A/D voltage comparator operation
ADM1 register:
ADTM01-ADTM00 bits ← 11; Hardware trigger wait mode
ADSCM1: one-shot conversion mode
ADTS1-ADTS0 bits ← 11: Set 12-bit interval timer interrupt signal as hardware trigger
ADM2 register:
ADTP0 ← 0; 10-bit resolution
ADREFP1-AVREF0 bits = 00B; Supply from VDD
AVREFM ← 0; Supply from VSS
ADUL register ← FFH; Set conversion result comparison upper limit value
ADLL register ← 80H; Set conversion result comparison lower limit value
ADS register ← 00H
```

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>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADCS</td>
<td>ADMD</td>
<td>FR2</td>
<td>FR1</td>
<td>FR0</td>
<td>LV1</td>
<td>LV0</td>
<td>ADCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Bit 6

<table>
<thead>
<tr>
<th>ADMD</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>FR2</th>
<th>FR1</th>
<th>FR0</th>
<th>LV1</th>
<th>LV0</th>
<th>Mode</th>
<th>Conversion Time</th>
<th>Conversion clock (fAD)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Normal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td>fCLK=1MHz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>fCLK=4MHz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>fCLK=8MHz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>fCLK=16MHz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>fCLK=24MHz</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Setting prohibited</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Setting prohibited</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Setting prohibited</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Normal</td>
<td>108 μs</td>
<td>fCLK/64</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td>108 μs</td>
<td>fCLK/32</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td>108 μs</td>
<td>fCLK/16</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td>108 μs</td>
<td>fCLK/8</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>108 μs</td>
<td>fCLK/5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td>108 μs</td>
<td>fCLK/4</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>108 μs</td>
<td>fCLK/2</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/G11 User’s Manual (hardware version) 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>
</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>
</tr>
<tr>
<td>ADTMD1</td>
<td>ADTMD0</td>
<td>ADSCM</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</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/G11 User’s Manual (hardware version) 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>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>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>

Bits 7-6

<table>
<thead>
<tr>
<th>ADREFP1</th>
<th>ADREFP0</th>
<th>Selection of the + side reference voltage source of the A/D converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Supplied from $V_{DD}$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Supplied from P20/AVREFP/ANI0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Supplied from the internal reference voltage (1.45 V)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Setting prohibited</td>
</tr>
</tbody>
</table>

Bit 5

<table>
<thead>
<tr>
<th>ADREFM</th>
<th>Selection of the - side reference voltage source of the A/D converter</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Supplied from $V_{SS}$</td>
</tr>
<tr>
<td>1</td>
<td>Supplied from P21/AVREFM/ANI1</td>
</tr>
</tbody>
</table>

Bit 3

<table>
<thead>
<tr>
<th>ADCRK</th>
<th>Checking the upper and lower limit conversion result values</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The interrupt signal (INTAD) is output when ADLL register ≤ ADCR register ≤ register ADUL.</td>
</tr>
<tr>
<td>1</td>
<td>The interrupt signal (INTAD) is output when ADCR register &lt; ADLL register, and ADUL register &lt; ADCR register.</td>
</tr>
</tbody>
</table>

Bit 2

<table>
<thead>
<tr>
<th>AWC</th>
<th>Specification of the 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>

Bit 0

<table>
<thead>
<tr>
<th>ADTYP</th>
<th>Selection of the A/D conversion resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10-bit resolution</td>
</tr>
<tr>
<td>1</td>
<td>8-bit resolution</td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/G11 User’s Manual (hardware version) 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>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>ADUL7</td>
<td>ADUL6</td>
<td>ADUL5</td>
<td>ADUL4</td>
<td>ADUL3</td>
<td>ADUL2</td>
<td>ADUL1</td>
<td>ADUL0</td>
</tr>
<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>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>ADLL7</td>
<td>ADLL6</td>
<td>ADLL5</td>
<td>ADLL4</td>
<td>ADLL3</td>
<td>ADLL2</td>
<td>ADLL1</td>
<td>ADLL0</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>
```

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>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>ADISS</td>
<td>0</td>
<td>0</td>
<td>ADS4</td>
<td>ADS3</td>
<td>ADS2</td>
<td>ADS1</td>
<td>ADS0</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>
```

Bits 7, 4-0

<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>P20/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>P21/ANI1 /AVREF pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>ANI2</td>
<td>P22/ANI2 pin</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>ANI3</td>
<td>P23/ANI3 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>ANI16</td>
<td>P01/ANI16 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>ANI17</td>
<td>P00/ANI17 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>ANI18</td>
<td>P33/ANI18 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>ANI19</td>
<td>P32/ANI19 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>ANI20</td>
<td>P31/ANI20 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>ANI21</td>
<td>P30/ANI21 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>ANI22</td>
<td>P56/ANI22 pin</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>PGAOUT (PGA output)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Temperature sensor 0 output</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Internal reference voltage output (1.45 V)</td>
<td></td>
</tr>
</tbody>
</table>

Note: Refer to the RL78/G11 User’s Manual (hardware version) for detailed explanations on how to set registers.
5.6.24 Status Transition HtoB

Figure 5.30 shows the status transition HtoB.

```
R_MAIN_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)
    P0.1, P0.0 bits ← 0: Set LED1, 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.

    R_MAIN_NOP_Loop()

return
```

Figure 5.30 Status Transition HtoB
5.6.25 Status Transition BtoC

Figure 5.31 and Figure 5.32 shows the status transition BtoC.

---

**Figure 5.31 Status Transition BtoC(1/2)**
Start count of 12-bit interval timer

MCS1 bit = 1?

Yes

Interval detection interrupt generated time?

Yes (branch when g_int_cnt = 1)

ITMC register ← 8001H

TMKAMK bit ← 1

Stop count of 12-bit interval timer

Stop high-speed on-chip 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

ITMC register ← 0000H

HIOSTOP bit ← 1

P5.6, P5.4 bits ← 1, P5.5 bit ← 0: Set LED3, LED5 to OFF and LED4 to ON

P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON

PIF0 bit ← 0

PMK0 bit ← 0

Execute NOP instruction continuously

To status transition error processing

Figure 5.32 Status Transition BtoC(2/2)
Figure 5.33 and Figure 5.34 show the status transition CtoD.

**R_MAIN_CtoD()**

1. **Disable external interrupt**
   - PMK0 bit ← 1
2. **Set oscillation stabilization time selection register**
   - OSTS register ← 07H
3. **X1 oscillator operation**
4. **Oscillation stability time elapsed?**
   - No
   - Yes (OSCT register branches at FFH)
5. **Select high-speed system clock as CPU/peripheral hardware clock**
   - MCM0 bit ← 1
6. **Clear interval detection interrupt request flag**
   - TMKAIF bit ← 0
7. **Enable interval detection interrupt**
   - TMKAMK bit ← 0
8. **Initialize interval detection interrupt generation counter**
9. **A**

**Figure 5.33 Status Transition CtoD(1/2)**
Start count of 12-bit interval timer

ITMC register ← 8001H

MCS bit = 1?

No

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

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)

P5.6, P5.4 bits ← 0, P5.5 bit ← 1: Set LED3, LED5 to ON and LED4 to OFF

Control LED lighting (CPU operation)

P0.1, P0.0 bits ← 0: Set LED1, 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

Figure 5.34 Status Transition CtoD(2/2)
5.6.27 Status Transition DtoE

Figure 5.35 and Figure 5.36 shows the status transition DtoE.

![Flowchart of Status Transition DtoE](image)

Figure 5.35 Status Transition DtoE(1/2)
Figure 5.36 Status Transition DtoE(2/2)
5.6.28 Status Transition EtoD

Figure 5.37 and Figure 5.38 shows the status transition EtoD.

```
R_MAIN_EtoD()

Disable external interrupt

Set oscillation stabilization time selection register

X1 oscillator operation

PMK0 bit ← 1

OSTS register ← 07H

MSTOP bit ← 0

Oscillation stabilization time elapsed?

No

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

CSS bit ← 0

Clear interval detection interrupt request flag

TMKAIF bit ← 0

Enable interval detection interrupt

TMKAMK bit ← 0

Initialize interval detection interrupt generation counter

Yes (branch when OSCT register is FFH)

A
```

**Figure 5.37 Status Transition EtoD(1/2)**
A

Start count of 12-bit interval timer

ITMC register ← 8005H

CLS bit = 0?

No

No

Yes (branch when g_int_cnt = 1)

R_MAIN_ERROR() To status transition error processing

Yes

Interval detection interrupt generated time?

TMKAMK bit ← 1

Stop count of 12-bit interval timer

ITMC register ← 0000H

Do not select low-speed on-chip oscillator clock as subsystem clock

SELLOSC bit ← 0

Control LED lighting (high-speed system clock)

P5.6, P5.4 bits ← 0, P5.5 bit ← 1: Set LED3, LED5 to ON and LED4 to OFF

Control LED lighting (CPU operation)

P0.1, P0.0 bits ← 0: Set LED1, 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 5.38 Status Transition EtoD(2/2)
5.6.29 Status Transition DtoC

Figure 5.39 and Figure 5.40 shows the status transition DtoC.

![Flowchart of Status Transition DtoC](image)

- **R_MAIN_DtoC()**
  - Disable external interrupt
  - Selection of middle-speed on-chip oscillator clock frequency
    - PMK0 bit ← 1
    - MOCODIV register ← 00H: Set the middle-speed on-chip oscillator frequency to 4MHz
  - Middle-speed on-chip oscillator operating
    - MIOEN bit ← 1
  - Clear interval detection interrupt request flag
    - TMKAIF bit ← 0
  - Enable interval detection interrupt
    - TMKAMK bit ← 0
  - Initialize counter of interval detection interrupt generation
  - Start count of 12-bit interval timer
    - ITMC register ← 8001H
  - Interval detection interrupt generated 1 times?
    - 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
        - MCM0 bit ← 0
      - Clear interval detection interrupt request flag
        - TMKAIF bit ← 0
      - Initialize counter of interval detection interrupt generation
    - No

Figure 5.39 Status Transition DtoC (1/2)
Figure 5.40  Status Transition DtoC(2/2)
5.6.30 Status Transition CtoI
Figure 5.41 shows the status transition CtoI.

```
R_MAIN_CtoI()

Control LED lighting (HALT mode)
P0.1 bit ← 0, P0.0 bit ← 1: Set LED1 to OFF and LED 2 to ON

Clear external interrupt request flag
PIF0 bit ← 0

HALT

return
```

Figure 5.41 Status Transition CtoI

5.6.31 Status Transition ItoC
Figure 5.42 shows the status transition ItoC.

```
R_MAIN_ItoC()

Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON

Clear external interrupt request flag
PIF0 bit ← 0

CPU operation (NOP instruction execution)
R_MAIN_NOP_Loop()

return
```

Figure 5.42 Status Transition ItoC
5.6.32 Status Transition CtoJ

Figure 5.43 shows the status transition CtoJ.

**Figure 5.43** Status Transition CtoJ

5.6.33 Status Transition JtoC

Figure 5.44 shows the status transition JtoC.

**Figure 5.44** Status Transition JtoC
5.6.34 Status Transition CtoK

Figure 5.45 shows the status transition CtoK.

```
R_MAIN_CtoK()
  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
    TMKAMK bit ← 1
  Disable interval detection interrupt
  Control LED lighting
    (SNOOZE mode)
    P0.1 bit ← 1, P0.0 bit ← 0: set LED1 to ON and LED2 to OFF
  Start 12-bit interval timer count
    ITMC register ← 0FFFH
  STOP
  return
```

Figure 5.45 Status Transition CtoK
5.6.35 Status Transition KtoC

Figure 5.46 shows the status transition KtoC.

```
R_MAIN_KtoC()

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)
P0.1, P0.0 bits ← 0: Set LED1, 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()

return

Execute NOP instruction continuously.
```

Figure 5.46 Status Transition KtoC
5.6.36 Status Transition CtoE
Figure 5.47 and Figure 5.48 shows the status transition CtoE.

![Flowchart for Status Transition CtoE](image)

- CtoE
  - 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
    - ITMC register ← 8001H

- CLS bit = 1?
  - No
    - Interval detection interrupt generated time?
      - Yes
        - Interval detection interrupt generated
          - Yes (branch when g_int_cnt = 1)
            - R_MAIN_ERROR()
          - No
            - To status transition error processing
      - No
        - To status transition error processing
    - Yes
      - To status transition error processing

Figure 5.47 Status Transition CtoE(1/2)
Disable interval detection interrupt
Stop 12-bit interval timer count
Stop middle-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)
R_MAIN_NOP_Loop()
return

TMKAMK bit ← 1
ITMC register ← 0000H
MIOEN bit ← 0
P5.6, P5.5 bit ← 0, P5.4 bits ← 1: Set LED3 to OFF and LED4, LED5 to ON
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
PIF0 bit ← 0
PMK0 bit ← 0
Execute NOP instruction continuously

Figure 5.48 Status Transition CtoE(2/2)
5.6.37 Status Transition EtoC

Figure 5.49 and Figure 5.50 shows the status transition EtoC.

**Figure 5.49 Status Transition EtoC(1/2)**
Start count of 12-bit interval timer

ITMC register ← 8001H

CLS bit = 0?

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

TMTC register ← 0000H

Stop count of 12-bit interval timer

ITMC register ← 0000H

Do not select low-speed on-chip oscillator clock as subsystem clock

SELLOSC bit ← 0

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

P5.6, P5.4 bits ← 1, P5.5 bit ← 0: Set LED3, LED5 to OFF and LED4 to ON

Control LED lighting (CPU operation)

P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON

Clear external interrupt request request flag

PIF0 bit ← 0

Enable external interrupt servicing

PMK0 bit ← 0

CPU operation (NOP instruction execution)

Execute NOP instruction continuously

R_MAIN_NOP_Loop()

return

Figure 5.50 Status Transition EtoC(2/2)
5.6.38 Status Transition CtoB

Figure 5.51 and Figure 5.52 shows the status transition CtoB.

---

**Figure 5.51 Status Transition CtoB(1/2)**

- Disable external interrupt
- High-speed on-chip oscillator operation
- Clear interval detection interrupt request flag
- Enable interval detection interrupt
- Initialize interval detection interrupt generation counter
- Start 12-bit interval timer count

**No**
- Interval detection interrupt generated 1 time?
  - Yes (branch when g_int_cnt = 1)
    - Stop 12-bit interval timer count
    - Select high-speed on-chip oscillator clock as CPU/peripheral hardware clock
    - Clear interval detection interrupt request flag
    - Initialize interval detection interrupt generation counter

**A**
CPU operation (NOP instruction execution)
\[\text{R\_MAIN\_NOP\_Loop()}\]

Figure 5.52  Status Transition CtoB(2/2)
5.6.39 Status Transition BtoE

Figure 5.53 and Figure 5.54 show the status transition BtoE.

![Flowchart]

**Figure 5.37 Status Transition BtoE (1/2)**
Figure 5.37  Status Transition BtoE(2/2)
5.6.40  Status Transition EtoN

Figure 5.55 shows the status transition EtoN.

5.6.41  Status Transition NtoE

Figure 5.56 shows the status transition NtoE.
5.6.42 Status Transition EtoB

Figure 5.57 and Figure 5.58 shows the status transition EtoB.

![Diagram of status transition EtoB](image)

**Figure 5.57 Status Transition EtoB(1/2)**
---

**Figure 5.58 Status Transition EtoB(2/2)**

- **Start count of 12-bit interval timer**
- **ITMC register ← 8001H**
  - **CLS 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 ← 0000H**
          - **TMKAMK bit ← 1**
    - **No**
      - **Stop count of 12-bit interval timer**
      - **Disable interval detection interrupt**
      - **Do not select low-speed on-chip oscillator clock as subsystem clock**
      - **Control LED lighting (high-speed on-chip oscillator clock)**
      - **Control LED lighting (CPU operation)**
      - **Clear external interrupt request request flag**
      - **Enable external interrupt servicing**
      - **CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()**
      - **return**
  - **SELLOSC bit ← 0**
  - **P5.6 bits ← 0, P5.5, P5.4 bits ← 1: Set LED3, LED4 to OFF and LED5 to ON**
  - **P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON**
  - **PIF0 bit ← 0**
  - **PMK0 bit ← 0**
  - **Execute NOP instruction continuously**

---
5.6.43 Status Transition End Processing

Figure 5.59 shows the flowchart for status transition end processing.

```
R_MAIN_END()

Disable interrupt

IE ← 0

Control LED lighting (OFF)

P5.6, P5.5, P5.4 bits ← 1: Set LED3, LED4, LED5 to OFF

Control LED lighting (OFF)

P0.1, P0.0 bits ← 1: Set LED1, LED2 to OFF

return
```

Figure 5.59 Status Transition End Processing
5.6.44 External Interrupt Servicing

Figure 5.60 shows the flowchart for external interrupt servicing.
5.6.45 12-bit Interval Timer Interrupt Servicing
Figure 5.61 shows the flowchart for 12-bit interval timer interrupt servicing.

```
// r_it_interrupt()
Increment interval detection interrupt generation counter by 1
return
```

**Figure 5.61 12-bit Interval Timer Interrupt Servicing**

5.6.46 A/D Conversion Completion Interrupt Servicing
Figure 5.62 shows the flowchart for A/D conversion completion interrupt servicing.

```
// r_adc_interrupt()
Stop use of SNOOZE mode function
AWC ← 0
return
```

**Figure 5.62 A/D Conversion Completion Interrupt Servicing**
6. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

7. Reference Documents

RL78/G11 User’s Manual: Hardware (R01UH0637J)
RL78 Family User’s Manual: Software (R01US0015J)

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>Nov. 30, 2016</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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.

2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.

3. Renesas Electronics does not assume any liability for infringement of 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. No licenses, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.

4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.

5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality." The recommended 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 systems; anti-disaster systems; anti-crime systems; and safety 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 implants etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable 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. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum ratings, operation supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages 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 its 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 be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, 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 use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.

9. Renesas Electronics products and technology may 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 should not use Renesas Electronics products or technology described in this document for any purpose relating to military use or by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.

10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.

11. This document may not be 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, or if you have any other inquiries.

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

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