Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. 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 license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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.
5. When exporting the 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. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. 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.
6. 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.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. 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 categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. 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 an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

“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.
“High Quality”:	Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
“Specific”:	Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.

8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating 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.
9. 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 them 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 system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. 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.
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.

Application Note

78K0S/Kx1+

Sample Program (Watchdog Timer)

131 ms Interval Runaway Detection

This document describes an operation overview of the sample program, as well as how to use the sample program and how to set and use the watchdog timer. In the sample program, a reset signal generated by an overflow of the watchdog timer is generated at a 50% chance of occurrence during a switch input, by setting the runaway detection time to 131 ms.

Target devices
78K0S/KA1+ microcontroller
78K0S/KB1+ microcontroller
78K0S/KU1+ microcontroller
78K0S/KY1+ microcontroller

CONTENTS

Document No. U18847EJ2V0AN00 (2nd edition)
Date Published September 2008 NS

- **The information in this document is current as of July, 2008. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**
- No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics. NEC Electronics assumes no responsibility for any errors that may appear in this document.
- NEC 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 NEC Electronics products listed in this document or any other liability arising from the use of such products. No license, express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Electronics or others.
- Descriptions of circuits, software and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software and information in the design of a customer's equipment shall be done under the full responsibility of the customer. NEC Electronics assumes no responsibility for any losses incurred by customers or third parties arising from the use of these circuits, software and information.
- While NEC Electronics endeavors to enhance the quality, reliability and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. To minimize risks of damage to property or injury (including death) to persons arising from defects in NEC Electronics products, customers must incorporate sufficient safety measures in their design, such as redundancy, fire-containment and anti-failure features.
- NEC Electronics products are classified into the following three quality grades: "Standard", "Special" and "Specific".

  The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

  "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.

  "Special": Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support).

  "Specific": Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems and medical equipment for life support, etc.

The quality grade of NEC Electronics products is "Standard" unless otherwise expressly specified in NEC Electronics data sheets or data books, etc. If customers wish to use NEC Electronics products in applications not intended by NEC Electronics, they must contact an NEC Electronics sales representative in advance to determine NEC Electronics' willingness to support a given application.

(Note)

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

(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

M8E 02. 11-1

CHAPTER 1 OVERVIEW

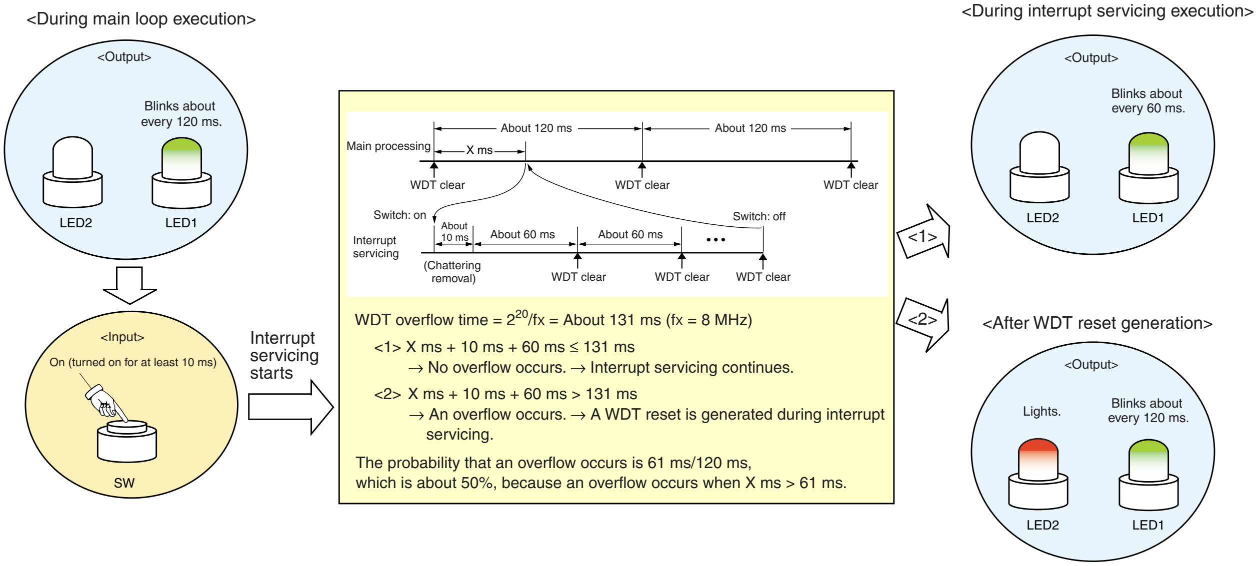
In this sample program, an example of using the watchdog timer (WDT) function is presented.

The system clock is set as the operation clock of the watchdog timer and the runaway detection time is set to about 131 ms.

After completion of the initial settings, LED1 out of the two LEDs (LED1, LED2) blinks about every 120 ms.

Either of the following operations is performed at a 50% chance of occurrence (see the figure below), depending on the switch input timing.

- LED1 blinks about every 60 ms during switch input by executing interrupt servicing.
- A reset signal is generated by an overflow of the watchdog timer during interrupt servicing. After reset release, LED2 lights and LED1 blinks about every 120 ms.



WDT overflow time = 2^{20}/fx = About 131 ms (fx = 8 MHz)

<1> X ms + 10 ms + 60 ms ≤ 131 ms
→ No overflow occurs. → Interrupt servicing continues.

<2> X ms + 10 ms + 60 ms > 131 ms
→ An overflow occurs. → A WDT reset is generated during interrupt servicing.

The probability that an overflow occurs is 61 ms/120 ms, which is about 50%, because an overflow occurs when X ms > 61 ms.

1.1 Main Contents of Initial Settings

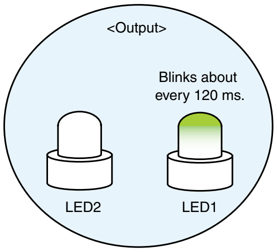
The contents of the initial settings are as follows.

- Selecting the high-speed internal oscillator (8 MHz (TYP.)) as the system clock source[Note]
- Setting so that oscillation of the low-speed internal oscillator can be stopped by using software[Note]
- Selecting the system clock (fx) for the watchdog timer operation clock, setting the overflow time to 2^{20}/fx (about 131 ms)
- Lighting LED2, when an internal reset signal is generated by the watchdog timer
- Setting V_{LVI} (low-voltage detection voltage) to 2.85 V ±0.15 V
- Generating an internal reset (LVI reset) signal when it is detected that V_{DD} is less than V_{LVI}, after V_{DD} (power supply voltage) becomes greater than or equal to V_{LVI}
- Setting the CPU clock frequency to 4 MHz
- Setting I/O ports
- Setting the valid edge of INTP1 (external interrupt) to the falling edge
- Enabling interrupt

Note This is set by using the option byte.

1.2 Contents Following the Main Loop

After completion of the initial settings, LED1 out of the two LEDs (LED1, LED2) blinks about every 120 ms, in the main loop.



Interrupt servicing is performed by detecting the falling edge of the INTP1 pin generated by switch input. If INTP1 is at high level (switch is turned off) after about 10 ms have elapsed since the falling edge of the INTP1 pin was detected, processing is identified as chattering and returned to the main loop. If INTP1 is at low level (switch is turned on) after about 10 ms have elapsed since edge detection, the following processing is advanced.

[Column] Chattering

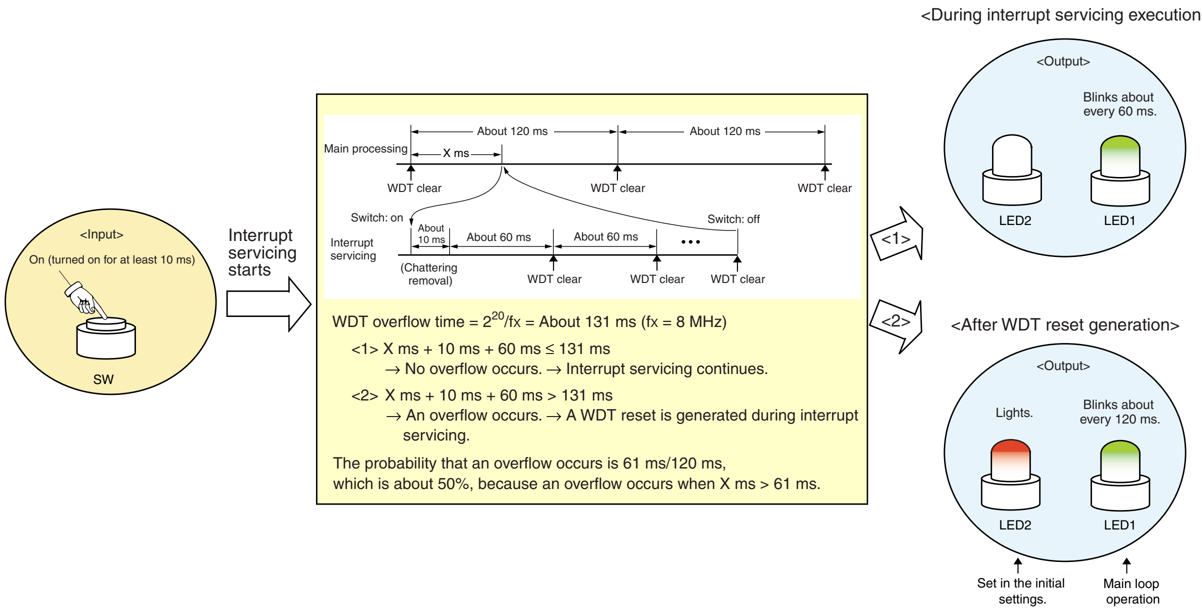
Chattering is a phenomenon in which the electric signal repeats turning on and off due to a mechanical flip-flop of the contacts, immediately after the switch has been pressed.

Either of the following operations is performed at a 50% chance of occurrence (see the figure below), depending on the switch input timing.

<1> If an overflow is not caused by the watchdog timer during interrupt servicing
Due to the execution of interrupt servicing, LED1 blinks about every 60 ms during switch input. When INTP1 goes to high level (switch is turned off), LED1 blinks about every 120 ms.

<2> If an overflow is caused by the watchdog timer during interrupt servicing
A reset signal is generated by the watchdog timer. After reset release, LED2 lights and LED1 blinks about every 120 ms.

Remark LED2 is turned off by a reset signal generated by other than the watchdog timer.



WDT overflow time = 2^{20}/fx = About 131 ms (fx = 8 MHz)

<1> X ms + 10 ms + 60 ms ≤ 131 ms
→ No overflow occurs. → Interrupt servicing continues.

<2> X ms + 10 ms + 60 ms > 131 ms
→ An overflow occurs. → A WDT reset is generated during interrupt servicing.

The probability that an overflow occurs is 61 ms/120 ms, which is about 50%, because an overflow occurs when X ms > 61 ms.

Cautions 1. In this sample program, the watchdog timer is not cleared at the beginning of interrupt servicing to generate a reset signal at a 50% chance of occurrence by the watchdog timer during interrupt servicing. In general use, the watchdog timer is cleared at the beginning and end of interrupt servicing so that no overflow occurs.

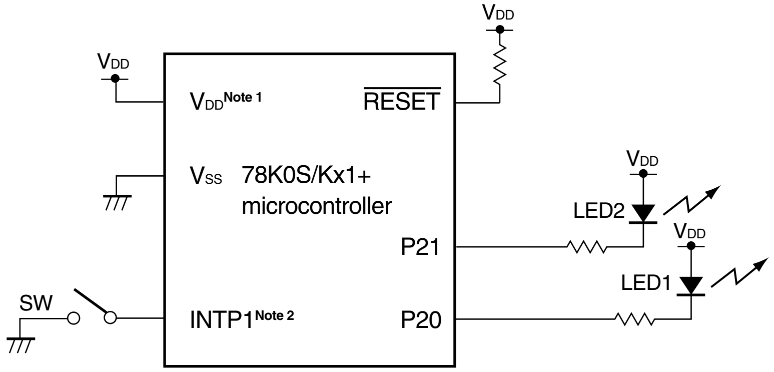
2. For cautions when using the device, refer to the user's manual of each product (78K0S/KU1+, 78K0S/KY1+, 78K0S/KA1+, 78K0S/KB1+).

CHAPTER 2 CIRCUIT DIAGRAM

This chapter describes a circuit diagram and the peripheral hardware to be used in this sample program.

2.1 Circuit Diagram

A circuit diagram is shown below.



Notes 1. Use this in a voltage range of 3.0 V ≤ V_{DD} ≤ 5.5 V.

2. INTP1/P43: 78K0S/KA1+ and 78K0S/KB1+ microcontrollers
INTP1/P32: 78K0S/KY1+ and 78K0S/KU1+ microcontrollers

Cautions 1. Connect the AV_{REF} pin directly to V_{DD} (only for the 78K0S/KA1+ and 78K0S/KB1+ microcontrollers).

2. Connect the AV_{SS} pin directly to GND (only for the 78K0S/KB1+ microcontroller).

3. Leave all unused pins open (unconnected), except for the pins shown in the circuit diagram and the AV_{REF} and AV_{SS} pins.

2.2 Peripheral Hardware

The peripheral hardware to be used is shown below.

(1) Switch (SW)

A switch is used as an input to control the lighting of an LED.

(2) LEDs (LED1, LED2)

The LEDs are used as outputs corresponding to switch inputs and reset signals generated by the watchdog timer.

CHAPTER 3 SOFTWARE

This chapter describes the file configuration of the compressed file to be downloaded, internal peripheral functions of the microcontroller to be used, and initial settings and operation overview of the sample program, and shows a flow chart.

3.1 File Configuration

The following table shows the file configuration of the compressed file to be downloaded.

File Name	Description	Compressed (*.zip) File Included	
		ZIP	PM 32
main.asm (Assembly language version)	Source file for hardware initialization processing and main processing of microcontroller	●Note	●Note
main.c (C language version)			
op.asm	Assembler source file for setting the option byte (sets the system clock source)	●	●
wdt.prw	Work space file for integrated development environment PM+		●
wdt.prj	Project file for integrated development environment PM+		●

Note "main.asm" is included with the assembly language version, and "main.c" with the C language version.

Remark ZIP : Only the source files are included.

PM 32 : The files to be used with integrated development environment PM+ are included.

3.2 Internal Peripheral Functions to Be Used

The following internal peripheral functions of the microcontroller are used in this sample program.

• Program runaway detection:	Watchdog timer
• $V_{DD} < V_{LVI}$ detection:	Low-voltage detector (LVI)
• Switch input:	INTP1[Note] (external interrupt)
• LED outputs:	P20, P21 (output ports)

Note INTP1/P43: 78K0S/KA1+ and 78K0S/KB1+ microcontrollers
INTP1/P32: 78K0S/KY1+ and 78K0S/KU1+ microcontrollers

3.3 Initial Settings and Operation Overview

In this sample program, setting of the watchdog timer and low-voltage detection function, selection of the clock frequency, and setting of the I/O ports, interrupts, and the like are performed in the initial settings.

After completion of the initial settings, LED1 out of the two LEDs (LED1, LED2) blinks about every 120 ms, in the main loop.

Interrupt servicing is performed by detecting the falling edge of the INTP1 pin generated by switch input. If INTP1 is at high level (switch is turned off) after about 10 ms have elapsed since the falling edge of the INTP1 pin was detected, processing is identified as chattering and returned to the main loop. If INTP1 is at low level (switch is turned on) after about 10 ms have elapsed since edge detection, the following processing is advanced.

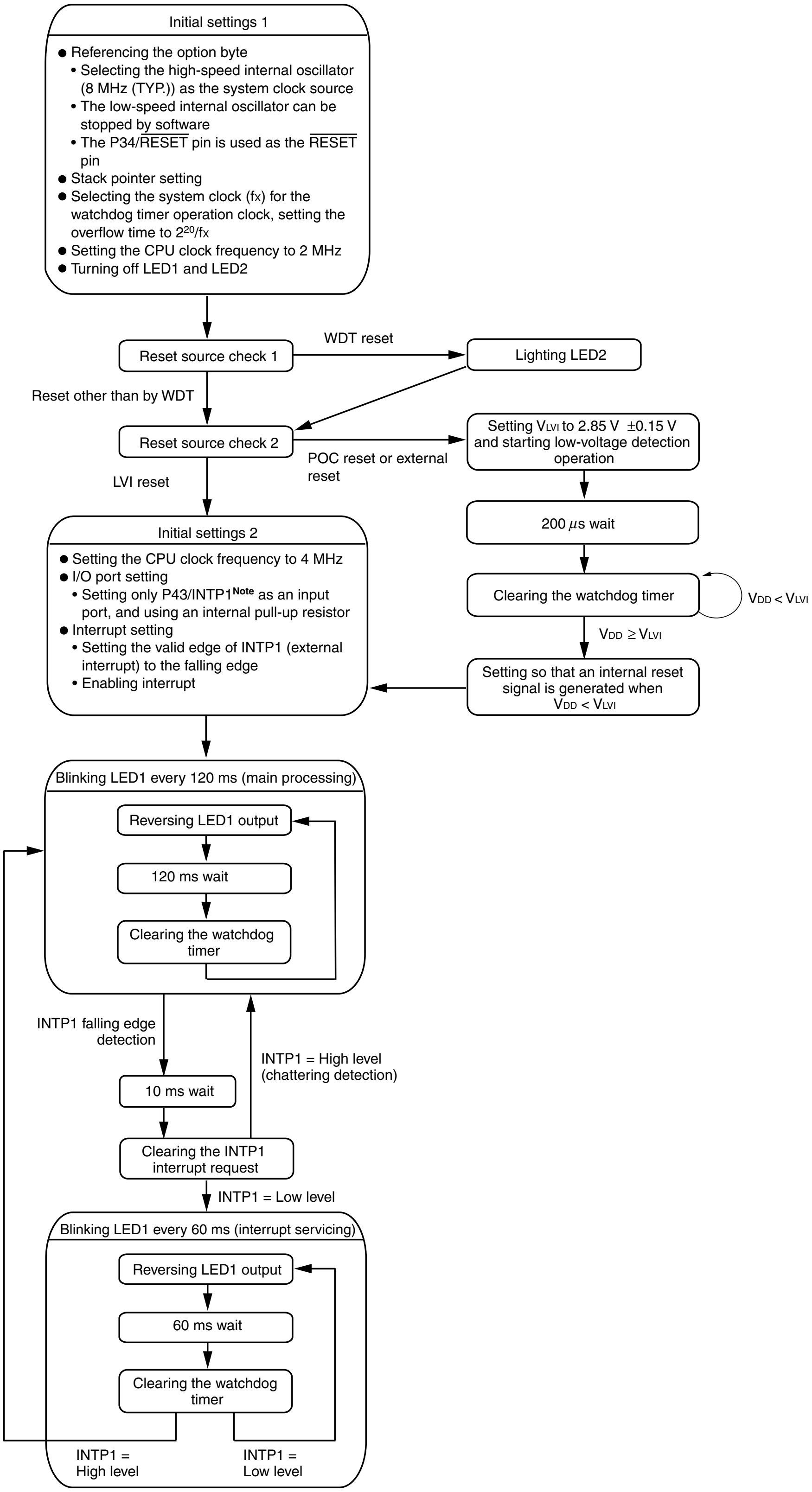
Either of the following operations is performed at a 50% chance of occurrence, depending on the switch input timing.

<1> If an overflow is not caused by the watchdog timer during interrupt servicing
Due to the execution of interrupt servicing, LED1 blinks about every 60 ms during switch input. When INTP1 goes to high level (switch is turned off), LED1 blinks about every 120 ms.

<2> If an overflow is caused by the watchdog timer during interrupt servicing
A reset signal is generated by the watchdog timer. After reset release, LED2 lights and LED1 blinks about every 120 ms.

Caution In this sample program, the count of the watchdog timer is not cleared at the beginning of interrupt servicing to generate a reset signal at a 50% chance of occurrence by the watchdog timer during interrupt servicing. In general use, the count of the watchdog timer is cleared at the beginning and end of interrupt servicing so that no overflow occurs.

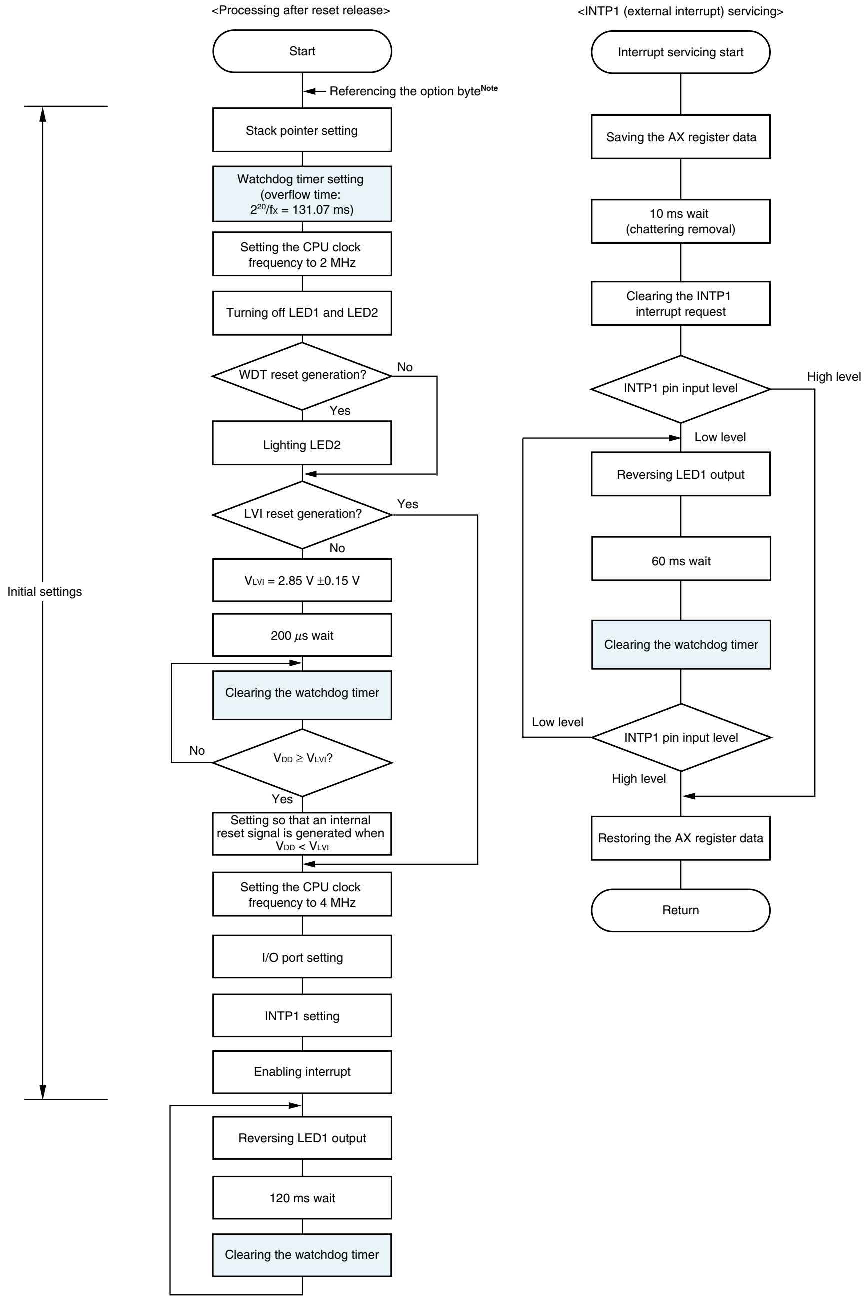
The details are described in the state transition diagram shown below.



Note INTP1/P43: 78K0S/KA1+ and 78K0S/KB1+ microcontrollers
INTP1/P32: 78K0S/KY1+ and 78K0S/KU1+ microcontrollers

3.4 Flow Chart

A flow chart for the sample program is shown below.



Note Referencing the option byte is automatically performed by the microcontroller after reset release. In this sample program, the following contents are set by referencing the option byte.

- Using the high-speed internal oscillation clock (8 MHz (TYP.)) as the system clock source
- The low-speed internal oscillator can be stopped by using software
- Using the P34/$\overline{\text{RESET}}$ pin as the $\overline{\text{RESET}}$ pin

Caution In this sample program, the watchdog timer is not cleared at the beginning of interrupt servicing to generate a reset signal at a 50% chance of occurrence by the watchdog timer during interrupt servicing. In general use, the watchdog timer is cleared at the beginning and end of interrupt servicing so that no overflow occurs.

CHAPTER 4 SETTING METHODS

This chapter describes the watchdog timer.

For other initial settings, refer to the **78K0S/Kx1+ Sample Program (Initial Settings) LED Lighting Switch Control Application Note**. For interrupt, refer to the **78K0S/Kx1+ Sample Program (Interrupt) External Interrupt Generated by Switch Input Application Note**. For low-voltage detection (LVI), refer to the **78K0S/Kx1+ Sample Program (Low-Voltage Detection) Reset Generation During Detection at Less than 2.7 V Application Note**.

For how to set registers, refer to the user's manual of each product (**78K0S/KU1+**, **78K0S/KY1+**, **78K0S/KA1+**, **78K0S/KB1+**).

For assembler instructions, refer to the **78K/0S Series Instructions User's Manual**.

4.1 Watchdog Timer (WDT) Setting

The watchdog timer is controlled by the following two types of registers.

- Watchdog timer mode register (WDTM)
- Watchdog timer enable register (WDTE)

The operation clock of the watchdog timer that can be selected depends on the oscillation control of the low-speed internal oscillator set by using the option byte.

(1) Setting the operation clock of the watchdog timer and overflow time

The watchdog timer mode register (WDTM) is used to set the operation clock of the watchdog timer and overflow time. Writing to WDTM can be performed only once after reset release.

Caution The operation clock of the watchdog timer and overflow time must be set in the initial settings.

Figure 4-1. Format of Watchdog Timer Mode Register (WDTM)

WDTM

0	1	1	WDCS4	WDCS3	WDCS2	WDCS1	WDCS0

Overflow time setting[Notes 1, 2]

			During Low-Speed Internal Oscillation Clock Operation	During System Clock Operation
0	0	0	$2^{11}/f_{RL}$	$2^{13}/f_X$
0	0	1	$2^{12}/f_{RL}$	$2^{14}/f_X$
0	1	0	$2^{13}/f_{RL}$	$2^{15}/f_X$
0	1	1	$2^{14}/f_{RL}$	$2^{16}/f_X$
1	0	0	$2^{15}/f_{RL}$	$2^{17}/f_X$
1	0	1	$2^{16}/f_{RL}$	$2^{18}/f_X$
1	1	0	$2^{17}/f_{RL}$	$2^{19}/f_X$
1	1	1	$2^{18}/f_{RL}$	$2^{20}/f_X$

Operation clock selection[Note 3]

0	0	Low-speed internal oscillation clock (f_{RL})
0	1	System clock (f_X)
1	x	Watchdog timer operation stop

Notes 1. If watchdog timer operation stop is selected, the overflow time setting is invalid (don't care).

2. The cycle is longest (WDCS2, WDCS1, WDCS0 = 1, 1, 1) when reset is released.

3. If the oscillation control of the low-speed internal oscillator is set to "Cannot be stopped" by using the option byte, the operation clock cannot be selected. The low-speed internal oscillation clock is selected regardless of the written value. Refer to **(3) Setting the oscillation control of the low-speed internal oscillator**.

Cautions 1. Set bits 7, 6, and 5 to 0, 1, and 1, respectively.

2. After reset is released, WDTM can be written only once. If writing is attempted a second time, an internal reset signal is generated. However, if "1" and "x" are set for WDCS4 and WDCS3, respectively, and the watchdog timer is stopped at the first write operation, no internal reset signal is generated even if a second write operation is executed.

Remark x: don't care

(2) Watchdog timer counter control

Writing “ACH” to the watchdog timer enable register (WDTE) clears the watchdog timer count and starts counting again.

Figure 4-2. Format of Watchdog Timer Enable Register (WDTE)

WDTE

Caution If a value other than “ACH” is written to WDTE, an internal reset signal is generated.

(3) Setting the oscillation control of the low-speed internal oscillator

The operation clock of the watchdog timer that can be used depends on the oscillation control of the low-speed internal oscillator set by using the option byte.

- Setting the oscillation of the low-speed internal oscillator to “Cannot be stopped”
  Operation clock: Only low-speed internal oscillation clock (operation clock cannot be selected)

- Setting the oscillation of the low-speed internal oscillator to “Can be stopped by using software”
  Operation clock: The low-speed internal oscillation clock, system clock, or watchdog timer operation stop can be selected.

Figure 4-3. Format of Option Byte (Only Oscillation Control of Low-Speed Internal Oscillator)

Address: 0080H

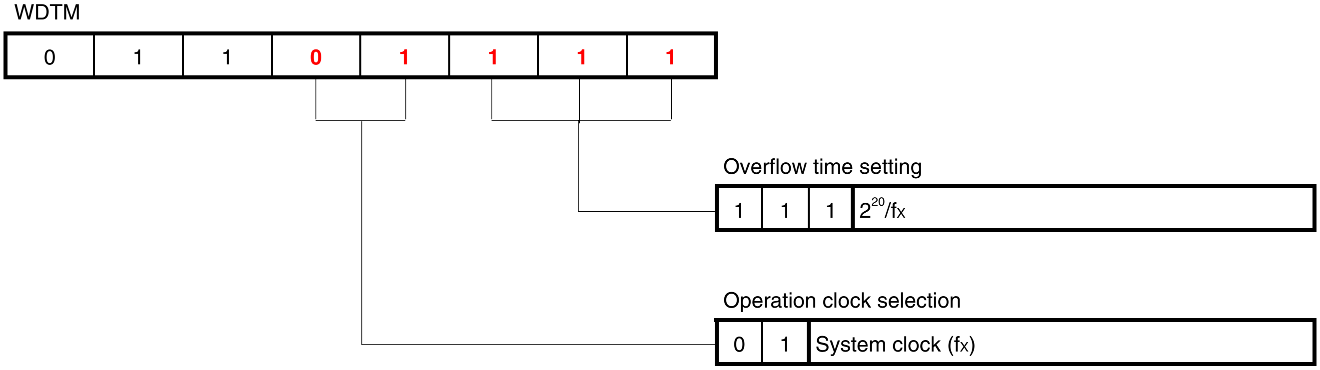
1	DEFOS TS1	DEFOS TS0	1	RMCE	OSCSE L1	OSCSE L0	LIOCP

Oscillation control of the low-speed internal oscillator

0	Can be stopped by using software
1	Cannot be stopped

Remark For option byte settings other than for the oscillation control of the low-speed internal oscillator, refer to the **78K0S/Kx1+ Sample Program (Initial Settings) LED Lighting Switch Control Application Note**.

[Example 1] Using the system clock (fx) as the watchdog timer operation clock, setting the overflow time to maximum cycle (2^{20}/fx) (same content as the sample program setting)



WDTM

0	1	1	0	1	1	1	1

Overflow time setting

1	1	1	2^{20}/fx

Operation clock selection

0	1	System clock (fx)

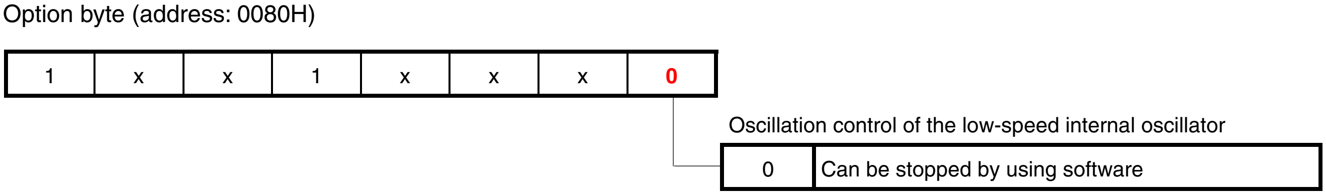
The WDTM setting value is “01101111 (bits 7, 6, and 5 must be set to 0, 1, and 1, respectively)”.

- Assembly language

```
MOV      WDTM,    #01101111B
```

- C language

```
WDTM = 0b01101111;
```



Option byte (address: 0080H)

1	x	x	1	x	x	x	0

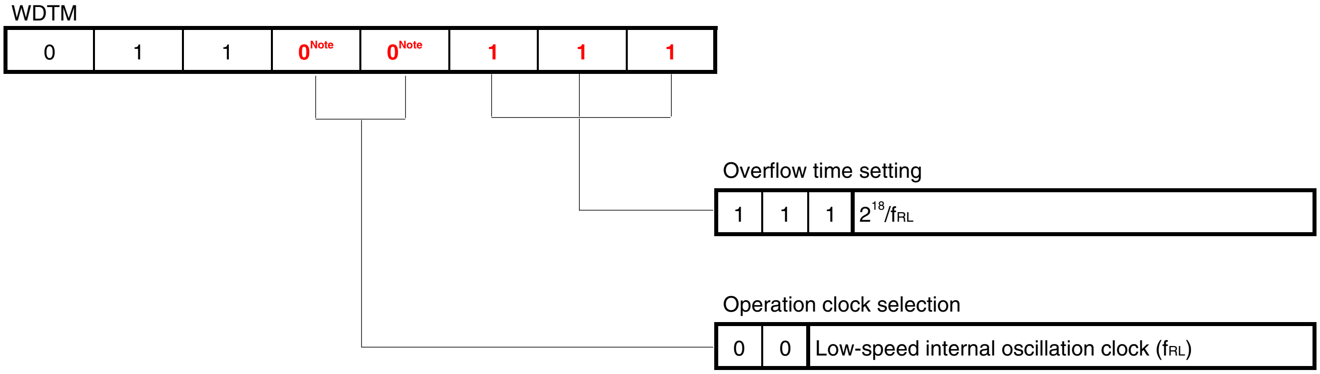
Oscillation control of the low-speed internal oscillator

0	Can be stopped by using software

The option byte setting value is “1xx1xxx0 (x: don’t care, bits 7 and 4 must be set to 1)”.
When the software is described together with the protect byte setting, the following results. (In the example below, bits 6, 5, and 1 are set to 0, and bits 3 and 2 are set to 1.)

```
OPBT   CSEG   AT     0080H
       DB     10011100B
       DB     11111111B
```

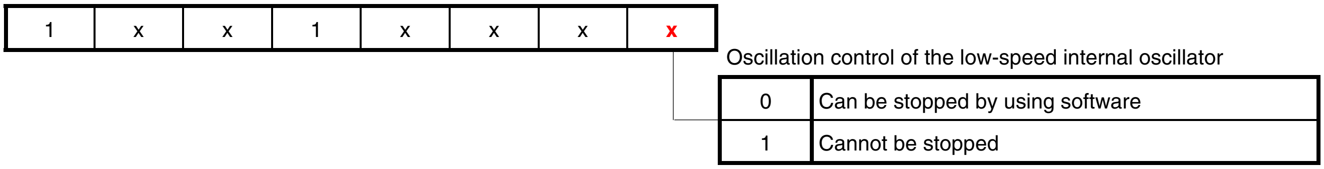
[Example 2] Using the low-speed internal oscillation clock (f_{RL}) as the watchdog timer operation clock, setting the overflow time to maximum cycle ($2^{18}/f_{RL}$) (same content as the WDTM value after reset release)



To use the watchdog timer as described above, the timer does not need to be set by a program, because the WDTM setting value is the same as that after reset release.

Note If the oscillation control of the low-speed internal oscillator is set to “Cannot be stopped” by using the option byte, the low-speed internal oscillation clock is selected, regardless of the written value.

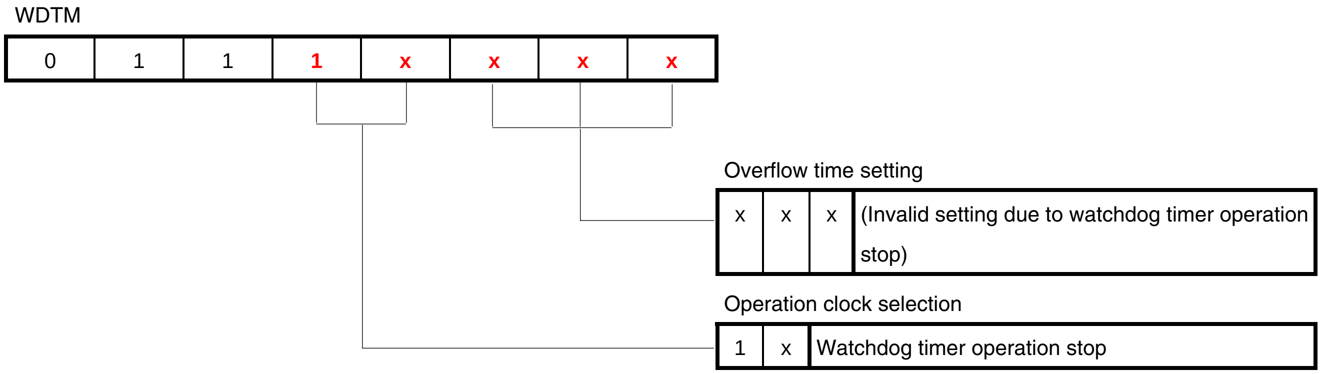
Option byte (address: 0080H)



The option byte setting value is “1xx1xxxx (x: don’t care, bits 7 and 4 must be set to 1)”.
When the software is described together with the protect byte setting, the following results. (In the example below, bits 6, 5, and 1 are set to 0, and bits 3, 2, and 0 are set to 1.)

```
OPBT   CSEG   AT     0080H
       DB     10011101B
       DB     11111111B
```

[Example 3] Stopping the watchdog timer



The WDTM setting value is “0111xxxx (x: don’t care, bits 7, 6, and 5 must be set to 0, 1, and 1, respectively)”. (In the example below, “x” of bit 3 and bits 2 to 0 is set to 0 and 1, respectively.)

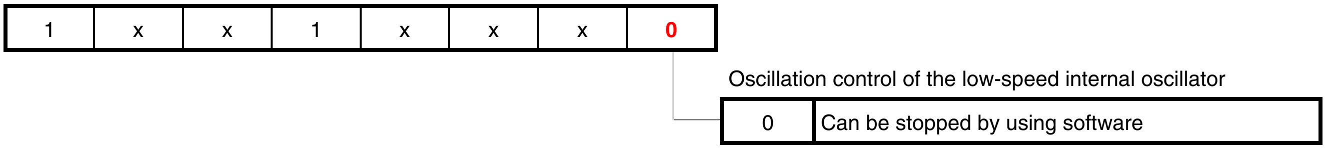
- Assembly language

```
MOV     WDTM,   #01110111B
```

- C language

```
WDTM = 0b01110111;
```

Option byte (address: 0080H)



The option byte setting value is “1xx1xxx0 (x: don’t care, bits 7 and 4 must be set to 1)”.
When the software is described together with the protect byte setting, the following results. (In the example below, bits 6, 5, and 1 are set to 0, and bits 3 and 2 are set to 1.)

```
OPBT    CSEG    AT      0080H
        DB      10011100B
        DB      11111111B
```

- Assembly language program example (same content as in [Example 1] described above and the sample program)

```
XMAIN  CSEG   UNIT
RESET_START:
       MOVW   AX,    #STACKTOP
       MOVW   SP,    AX             ; Set the stack pointer

       MOV    WDTM,  #01101111B     ; WDT overflow time = 2^20/fx = 131.07 ms
                     •
                     •
                     •
       MOV    A,     RESF           ; Read the reset source
       BF     A.4,   $CHECK_LVI     ; Go to CHECK_LVI if not a reset by WDT
       MOV    P2,    #00000001B     ; Light LED2

CHECK_LVI:
       BT     A.0,   $SET_CLOCK     ; Omit subsequent LVI-related processing and go to SET_CLOCK
during LVI reset
       MOV    LVIS,  #00000111B     ; Set the low-voltage detection level (VLVI) to 2.85 V +-0.15 V
       SET1   LVION                 ; Enable the low-voltage detector operation
       MOV    A,     #40            ; Assign the 200 us wait count value

WAIT_200US:
       DEC    A
       BNZ    $WAIT_200US           ; 0.5[us/clk] × 10[clk] × 40[count] = 200[us]

WAIT_LVI:
       MOV    WDTE,  #0ACH          ; Clear the watchdog timer
       BT     LVIF,  $WAIT_LVI      ; Branch if VDD < VLVI
       SET1   LVIMD                 ; Set so that an internal reset signal is generated when VDD < VLVI
                     •
                     •
                     •
MAIN_LOOP:
       XOR    P2,    #00000001B     ; Reverse output of LED1
       MOV    CNT120,       #120    ; Assign the 120 ms wait count value
WAIT_120MS:
       CALL   !WAIT_1MS             ; Subroutine call for a 1 ms wait
       DBNZ   CNT120, $WAIT_120MS   ; 1[ms] × 120[count] = 120[ms]
       MOV    WDTE,  #0ACH          ; Clear the watchdog timer
       BR     $MAIN_LOOP            ; Go to the MAIN_LOOP
                     •
                     •
                     •
```

Setting the WDT overflow time and operation clock

Clearing WDT before overflow occurrence and restarting counting

Clearing WDT before overflow occurrence and restarting counting

Remark The above-mentioned wait time (200 μs) is calculated with f_{CPU} (CPU clock frequency) being 2 MHz, as done in the sample program.

- C language program example (same content as in [Example 1] described above and the sample program)

```
void hdwinit(void){
        unsigned char ucCnt200us;  /* 8-bit variable for 200 us wait */
        WDTM  = 0b01101111;        /* WDT overflow time = 2^20/fx = 131.07 ms */
                •
                •
                •
```

Setting the WDT overflow time and operation clock

```
        /* Check the reset source */
        ucRESF = RESF;             /* Read the reset source */

        if (ucRESF.4){             /* A reset generated by WDT */
                P2 = 0b00000001;   /* Light LED2 */
        }

        if (!ucRESF.0){        /* Omit subsequent LVI-related processing during LVI reset */

                /* Set low-voltage detection */
                LVIS  = 0b00000111; /* Set the low-voltage detection level (VLVI) to 2.85 V +-0.15 V
*/
                LVION = 1;         /* Enable the low-voltage detector operation */

                for (ucCnt200us = 0; ucCnt200us < 9; ucCnt200us++){  /* Wait of about 200 us */
                        NOP();
                }

                while (LVIF){      /* Wait for VDD >= VLVI */
                        WDTE = 0xAC; /* Clear the watchdog timer */
                }

                LVIMD = 1;         /* Set so that an internal reset signal is generated when VDD < VLVI */
        }
                •
                •
                •
```

Clearing WDT before overflow occurrence and restarting counting

```
void main(void){
        unsigned int unCnt120ms;   /* 16-bit variable for 120 ms wait */

        EI();                      /* Enable vector interrupt */

        while (1){
                P2 ^= 0b00000001;  /* Reverse output of LED1 */

                for (unCnt120ms = 0; unCnt120ms < 6666; unCnt120ms++){
                                   /* Wait of about 120 ms */
                        NOP();
                }

                WDTE  = 0xAC;      /* Clear the watchdog timer */
        }
}
                •
                •
```

Clearing WDT before overflow occurrence and restarting counting

Remark The above-mentioned wait time (200 μs) is calculated with f_{CPU} (CPU clock frequency) being 2 MHz, as done in the sample program.

CHAPTER 5 OPERATION CHECK USING THE DEVICE

This chapter describes the flow from building to the operation check using the device, using the downloaded sample program.

<R> 5.1 Building the Sample Program

This section describes how to build a sample program by using the assembly language sample program (source program + project file) downloaded by clicking the icon. See the **78K0S/Kx1+ Sample Program Startup Guide Application Note** for how to build other downloaded programs.

For the details of how to operate PM+, refer to the **PM+ Project Manager User's Manual**.

[Column] Build errors

Change the compiler option setting according to the following procedure when the error message "A006 File not found 'C:\NECTOOLS32\LIB78K0S\s0sl.rel'" or "*** ERROR F206 Segment '@@DATA' can't allocate to memory - ignored." is displayed, when building with PM+.

<1> Select [Compiler Options] from the [Tool] menu.
<2> The [Compiler Options] dialog box will be displayed. Select the [Startup Routine] tab.
<3> Uncheck the [Using Fixed Area of Standard Library] check box. (Leave the other check boxes as they are.)

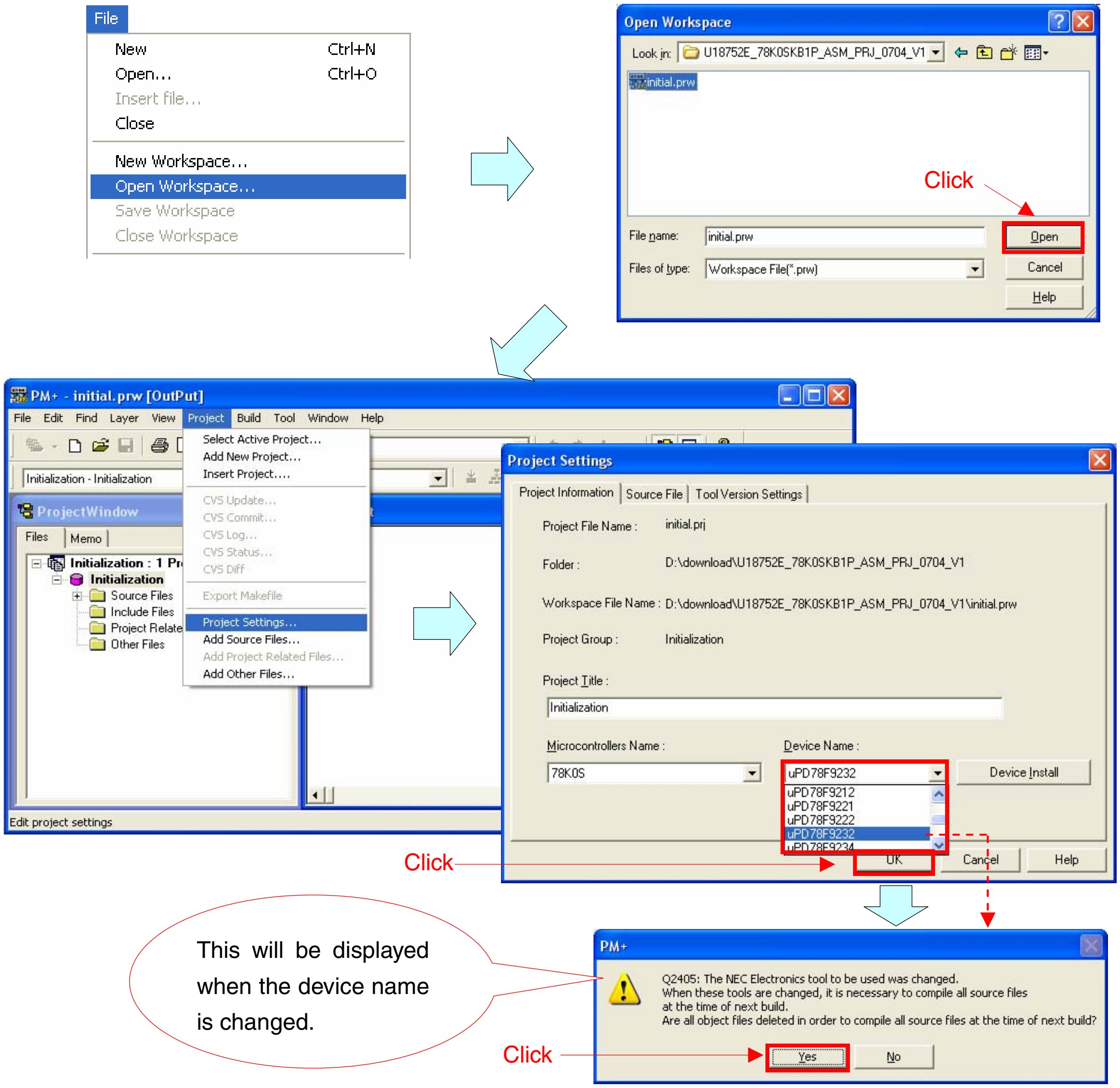
A RAM area of 118 bytes that has been secured as a fixed standard library area will be enabled for use when the [Using Fixed Area of Standard Library] check box is unchecked; however, the standard libraries (such as the getchar function and malloc function) will be disabled for use.
The [Using Fixed Area of Standard Library] check box is unchecked by default when the file that has been downloaded by clicking the icon is used in this sample program.

(1) Start PM+.

(2) Select “wdt.prw” by clicking [Open Workspace] from the [File] menu and click [Open]. A workspace into which the source file will be automatically read will be created.

(3) Select [Project Settings] from the [Project] menu. When the [Project Settings] window opens, select the name of the device to be used (the device with the largest ROM or RAM size will be selected by default), and click [OK].

Remark Screenshots of the Sample Program (Initial Settings) LED Lighting Switch Control are shown below.



Look in: U18752E_78K0SKB1P_ASM_PRJ_0704_V1

File name: initial.prw

Files of type: Workspace File(*.prw)

Project File Name : initial.prj

Folder : D:\download\U18752E_78K0SKB1P_ASM_PRJ_0704_V1

Workspace File Name : D:\download\U18752E_78K0SKB1P_ASM_PRJ_0704_V1\initial.prw

Project Group : Initialization

Project Title : Initialization

Microcontrollers Name : 78K0S

Device Name : uPD78F9232

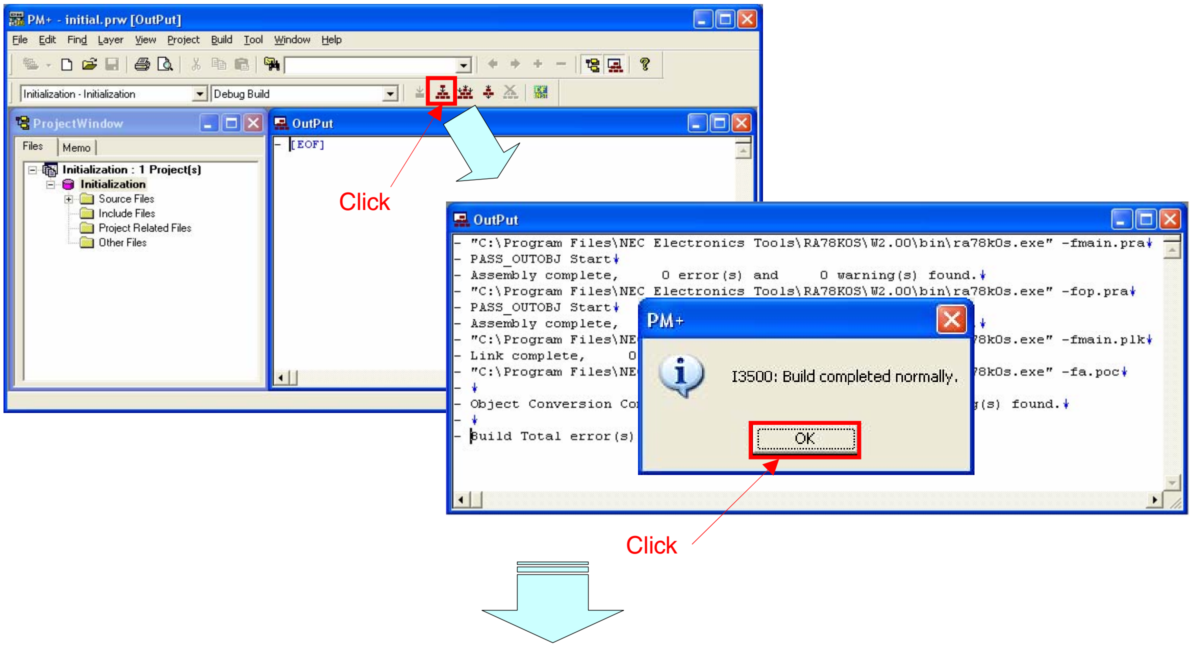
This will be displayed when the device name is changed.

Q2405: The NEC Electronics tool to be used was changed.
When these tools are changed, it is necessary to compile all source files at the time of next build.
Are all object files deleted in order to compile all source files at the time of next build?

(4) Click ([Build] button). When the source files are built normally, the message “I3500: Build completed normally.” will be displayed.

(5) Click the [OK] button in the message dialog box. A HEX file for flash memory writing will be created.

Remark Screenshots of the Sample Program (Initial Settings) LED Lighting Switch Control are shown below.



```
- "C:\Program Files\NEC Electronics Tools\RA78K0S\W2.00\bin\ra78k0s.exe" -fmain.pra↓
- PASS_OUTOBJ Start↓
- Assembly complete,     0 error(s) and     0 warning(s) found.↓
- "C:\Program Files\NEC Electronics Tools\RA78K0S\W2.00\bin\ra78k0s.exe" -fop.pra↓
- PASS_OUTOBJ Start↓
- Assembly complete,                                                   ↓
- "C:\Program Files\NE                                   78k0s.exe" -fmain.plk↓
- Link complete,     0
- "C:\Program Files\NE                                   78k0s.exe" -fa.poc↓
- ↓
- Object Conversion Co                                   (s) found.↓
- ↓
- Build Total error(s)
```

I3500: Build completed normally.

A HEX file for flash memory writing will be generated.

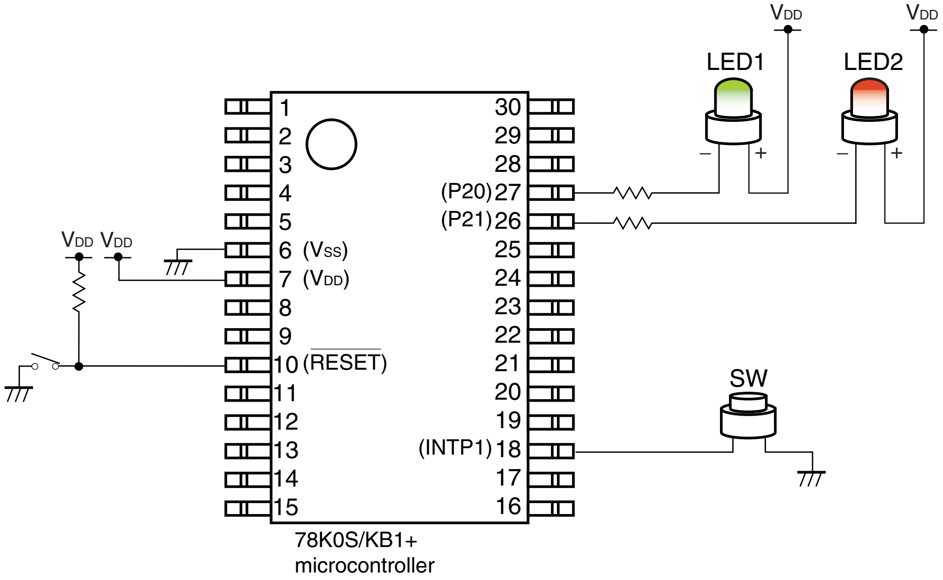
5.2 Operation with the Device

This section describes an example of an operation check using the device.

The HEX file generated by executing build can be written to the flash memory of the device.

For how to write to the flash memory of the device, refer to the **Flash Programming Manual (Basic) MINICUBE2 version** of each product **(78K0S/KU1+, 78K0S/KY1+, 78K0S/KA1+, 78K0S/KB1+)**.

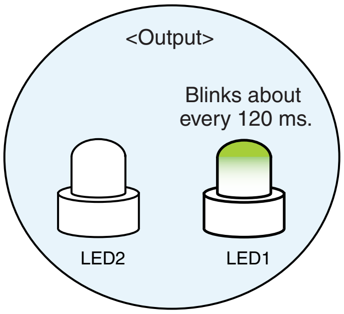
An example of how to connect the device and peripheral hardware (switch and LEDs) to be used is shown below.



An operation example of the device to which this sample program has been written is described below.

(1) Operation before pressing the switch (operation of main processing)

LED1 blinks about every 120 ms.



(2) Operation after pressing the switch (operation after interrupt servicing)

If the switch is pressed for less than 10 ms, the switch input is identified as chattering and operation is returned to that described in (1).

If the switch is pressed for longer than 10 ms, either of the following operations is performed at a 50% chance of occurrence (see the figure below), depending on the timing at which the switch is pressed.

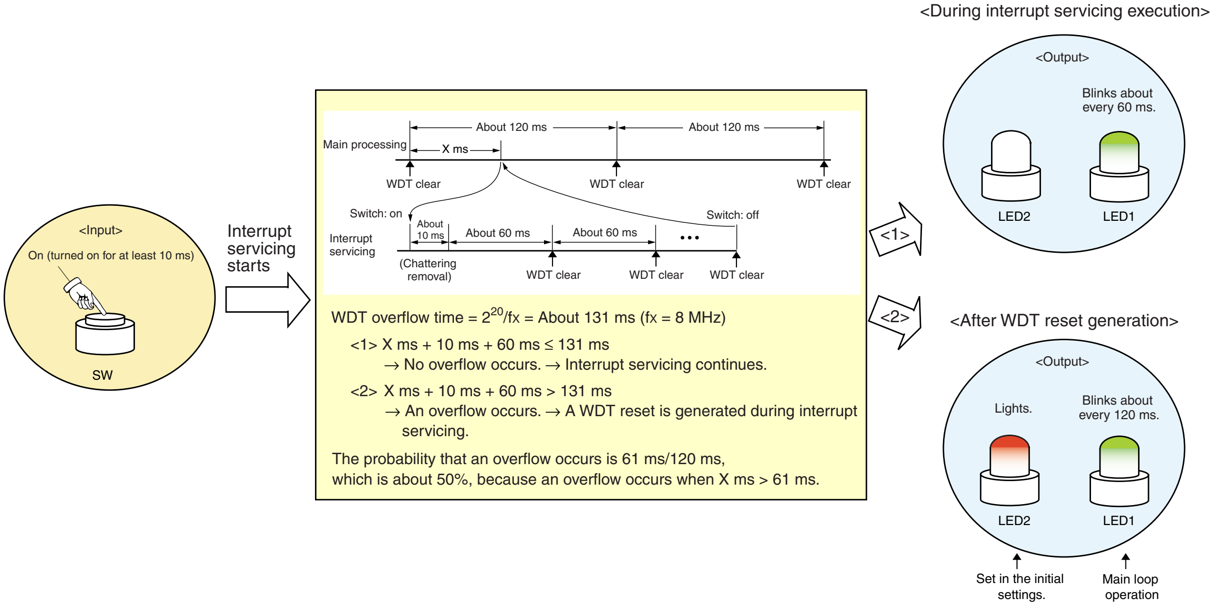
<1> If an overflow is not caused by the watchdog timer after the switch is pressed

When the switch is turned on (switch is kept pressed), LED1 blinks about every 60 ms. When the switch is turned off, LED1 blinks about every 120 ms.

<2> If an overflow is caused by the watchdog timer after the switch is pressed

A reset signal is generated by the watchdog timer. After reset release, LED2 lights and LED1 blinks about every 120 ms.

Remark LED2 is turned off by a reset signal generated by other than the watchdog timer.



WDT overflow time = 2^{20}/fx = About 131 ms (fx = 8 MHz)

<1> X ms + 10 ms + 60 ms ≤ 131 ms
→ No overflow occurs. → Interrupt servicing continues.

<2> X ms + 10 ms + 60 ms > 131 ms
→ An overflow occurs. → A WDT reset is generated during interrupt servicing.

The probability that an overflow occurs is 61 ms/120 ms, which is about 50%, because an overflow occurs when X ms > 61 ms.

CHAPTER 6 RELATED DOCUMENTS

Document Name		Japanese/English
78K0S/KU1+ User's Manual		PDF
78K0S/KY1+ User's Manual		PDF
78K0S/KA1+ User's Manual		PDF
78K0S/KB1+ User's Manual		PDF
78K/0S Series Instructions User's Manual		PDF
RA78K0S Assembler Package User's Manual	Language	PDF
	Operation	PDF
CC78K0S C Compiler User's Manual	Language	PDF
	Operation	PDF
PM+ Project Manager User's Manual		PDF
SM+ System Simulator Operation User's Manual		PDF
<R> Flash Programming Manual (Basic) MINICUBE2 version	78K0S/KU1+	PDF
	78K0S/KY1+	PDF
	78K0S/KA1+	PDF
	78K0S/KB1+	PDF
78K0S/Kx1+ Application Note	Sample Program Startup Guide	PDF
	Sample Program (Initial Settings) LED Lighting Switch Control	PDF
	Sample Program (Interrupt) External Interrupt Generated by Switch Input	PDF
	Sample Program (Low-Voltage Detection) Reset Generation During Detection at Less than 2.7 V	PDF

APPENDIX A PROGRAM LIST

As a program list example, the 78K0S/KB1+ microcontroller source program is shown below.

- main.asm (Assembly language version)

```
;*****************************************************************************
;
;     NEC Electronics     78K0S/KB1+
;
;*****************************************************************************
;     78K0S/KB1+  Sample program
;*****************************************************************************
;     Watchdog timer
;*****************************************************************************
;<<History>>
;     2007.6.--   Release
;*****************************************************************************
;
;<<Overview>>
;
;This sample program presents an example of using the watchdog timer function.
;The overflow time of the watchdog timer is set to 131.07 ms by counting the 8
;MHz fx and runaway detection is performed.  (fx: system clock source)
;After completion of the initial settings, LED1 blinks every 120 ms and blinks
;every 60 ms when the switch is pressed.
;Here, a reset is generated by the watchdog timer at a 50% chance of
;occurrence, depending on the timing at which SW is pressed, and LED2 lights,
;because the watchdog timer is cleared about 70 ms after the interrupt of the
;SW input was generated.
;# Normally, an overflow should be avoided by clearing the watchdog timer at
;  the beginning and end of an interrupt.
;
;
;  <Principal setting contents>
;
;  - Set the low-voltage detection voltage (VLVI) to 2.85 V +-0.15 V
;  - Generate an internal reset signal (low-voltage detector) when VDD < VLVI
after VDD >= VLVI
;  - Set the CPU clock frequency to 4 MHz
;  - Select fx as the watchdog timer (WDT) count clock and
;    set the overflow time to 131.07 ms
;  - Clear the watchdog timer about every 120 ms in the main loop
;  - Clear the watchdog timer about 70 ms after the interrupt of the switch
input was generated and clear the watchdog timer about every 60 ms if the
switch input continues thereafter
;  - Set the valid edge of external interrupt INTP1 to falling edge
;  - Set the chattering removal time during switch input to 10 ms
;
;
;  <SW input and LED1>
;
;    +----------------------------+
;    |  SW   |        LED1        |
;    | (P43) |       (P20)        |
;    |-------|--------------------|
```

```
;    |  OFF  | Blink every 120 ms |
;    |-------|--------------------|
;    |  ON   | Blink every 60 ms  |#
;    +----------------------------+
;   # A reset is generated by WDT at a 50% chance of occurrence when SW is
turned on.
;     In that case, LED1 blinks every 120 ms even if SW is turned on.
;
;
;  <RESET source and LED2>
;
;    +------------------------+
;    |  RESET Source  |  LED2 |
;    |                | (P21) |
;    |----------------|-------|
;    | Other than WDT |  OFF  |
;    |----------------|-------|
;    | WDT            |  ON   |
;    +------------------------+
;
;
;<<I/O port settings>>
;
;  Input: P43
;  Output: P00-P03, P20-P23, P30-P33, P40-P42, P44-P47, P120-P123, P130
;  # All unused ports are set as the output mode.
;
;****************************************************************************


;============================================================================
;
;     Vector table
;
;============================================================================
XVCT  CSEG  AT    0000H
      DW    RESET_START       ;(00) RESET
      DW    RESET_START       ;(02) --
      DW    RESET_START       ;(04) --
      DW    RESET_START       ;(06) INTLVI
      DW    RESET_START       ;(08) INTP0
      DW    INTERRUPT_P1      ;(0A) INTP1
      DW    RESET_START       ;(0C) INTTMH1
      DW    RESET_START       ;(0E) INTTM000
      DW    RESET_START       ;(10) INTTM010
      DW    RESET_START       ;(12) INTAD
      DW    RESET_START       ;(14) --
      DW    RESET_START       ;(16) INTP2
      DW    RESET_START       ;(18) INTP3
      DW    RESET_START       ;(1A) INTTM80
      DW    RESET_START       ;(1C) INTSRE6
      DW    RESET_START       ;(1E) INTSR6
      DW    RESET_START       ;(20) INTST6

;============================================================================
;
;     Define the RAM
;
;============================================================================
```

```
XRAM  DSEG  SADDR
CNT10:      DS    1                 ; For 10 ms wait
CNT60:      DS    1                 ; For 60 ms wait
CNT120:     DS    1                 ; For 120 ms wait

;==============================================================================
;
;     Define the memory stack area
;
;==============================================================================
XSTK  DSEG  AT    0FEE0H
STACKEND:
      DS    20H               ; Memory stack area = 32 bytes
STACKTOP:                     ; Start address of the memory stack area = FF00H

;******************************************************************************
;
;     Initialization after RESET
;
;******************************************************************************
XMAIN CSEG  UNIT
RESET_START:
;------------------------------------------------------------------------------
;     Initialize the stack pointer
;------------------------------------------------------------------------------
      MOVW  AX,   #STACKTOP
      MOVW  SP,   AX          ; Set the stack pointer

;------------------------------------------------------------------------------
;     Initialize the watchdog timer + detect low-voltage + set the clock
;------------------------------------------------------------------------------
;-----  Initialize the watchdog timer  -----
      MOV   WDTM, #01101111B  ; WDT overflow time = 2^20/fx = 131.07 ms

;-----  Set the clock <1>  -----
      MOV   PCC,  #00000000B  ; The clock supplied to the CPU (fcpu) = fxp (=
fx/4 = 2 MHz)
      MOV   LSRCM,      #00000001B  ; Stop the oscillation of the low-speed
internal oscillator

;-----  Initialize the port 2  -----
      MOV   P2,   #00000011B  ; Set output latches of P20, P21 as high (turn
off LED1, LED2) and P22, P23 as low
      MOV   PM2,  #11110000B  ; Set P20-P23 as output mode

;-----  Check the reset source  -----
      MOV   A,    RESF        ; Read the reset source
      BF    A.4,  $CHECK_LVI  ; Go to CHECK_LVI if not a reset by WDT
      MOV   P2,   #00000001B  ; Light LED2

CHECK_LVI:
      BT    A.0,  $SET_CLOCK  ; Omit subsequent LVI-related processing and go
to SET_CLOCK during LVI reset

;-----  Set low-voltage detection  -----
      MOV   LVIS, #00000111B  ; Set the low-voltage detection level (VLVI) to
2.85 V +-0.15 V
      SET1  LVION             ; Enable the low-voltage detector operation
```

```
      MOV   A,    #40         ; Assign the 200 us wait count value
;-----  200 us wait  -----
WAIT_200US:
      DEC   A
      BNZ   $WAIT_200US       ; 0.5[us/clk] x 10[clk] x 40[count] = 200[us]

;-----  Wait for VDD >= VLVI  -----
WAIT_LVI:
      MOV   WDTE, #0ACH       ; Clear the watchdog timer
      BT    LVIF, $WAIT_LVI   ; Branch if VDD < VLVI

      SET1  LVIMD             ; Set so that an internal reset signal is
generated when VDD < VLVI

;-----  Set the clock <2>  -----
SET_CLOCK:
      MOV   PPCC, #00000001B  ; The clock supplied to the peripheral hardware
(fxp) = fx/2 (= 4 MHz)
                              ; -> The clock supplied to the CPU (fcpu) = fxp
= 4 MHz

;-----------------------------------------------------------------------------
;     Initialize the port 0
;-----------------------------------------------------------------------------
      MOV   P0,   #00000000B  ; Set output latches of P00-P03 as low
      MOV   PM0,  #11110000B  ; Set P00-P03 as output mode

;-----------------------------------------------------------------------------
;     Initialize the port 3
;-----------------------------------------------------------------------------
      MOV   P3,   #00000000B  ; Set output latches of P30-P33 as low
      MOV   PM3,  #11110000B  ; Set P30-P33 as output mode

;-----------------------------------------------------------------------------
;     Initialize the port 4
;-----------------------------------------------------------------------------
      MOV   P4,   #00000000B  ; Set output latches of P40-P47 as low
      MOV   PU4,  #00001000B  ; Connect on-chip pull-up resistor to P43
      MOV   PM4,  #00001000B  ; Set P43 as input mode, P40-P42 and P44-P47 as
output mode

;-----------------------------------------------------------------------------
;     Initialize the port 12
;-----------------------------------------------------------------------------
      MOV   P12,  #00000000B  ; Set output latches of P120-P123 as low
      MOV   PM12, #11110000B  ; Set P120-P123 as output mode

;-----------------------------------------------------------------------------
;     Initialize the port 13
;-----------------------------------------------------------------------------
      MOV   P13,  #00000001B  ; Set output latch of P130 as high

;-----------------------------------------------------------------------------
;     Set the interrupt
;-----------------------------------------------------------------------------
      MOV   INTM0,      #00000000B  ; Set the valid edge of INTP1 to falling
edge
      CLR1  PIF1              ; Clear invalid interrupt requests in advance
      CLR1  PMK1              ; Release the INTP1 interrupt mask
```

```
      EI                        ; Enable vector interrupt

;*****************************************************************************
;
;     Main loop
;
;*****************************************************************************
MAIN_LOOP:
;-----  Reverse LED1  -----
      XOR   P2,   #00000001B  ; Reverse output of LED1

      MOV   CNT120,     #120  ; Assign the 120 ms wait count value
;-----  120 ms wait  -----
WAIT_120MS:
      CALL  !WAIT_1MS         ; Subroutine call for a 1 ms wait
      DBNZ  CNT120,     $WAIT_120MS ; 1[ms] x 120[count] = 120[ms]

;-----  Clear the watchdog timer  -----
      MOV   WDTE, #0ACH       ; Clear the watchdog timer
      BR    $MAIN_LOOP        ; Go to the MAIN_LOOP

;*****************************************************************************
;
;     External interrupt INTP1
;
;*****************************************************************************
INTERRUPT_P1:
      PUSH  AX                ; Save the AX register data to the stack

      MOV   CNT10,      #10   ; Assign the 10 ms wait count value
;-----  10 ms wait to handle chattering  -----
WAIT_10MS:
      CALL  !WAIT_1MS         ; Subroutine call for a 1 ms wait
      DBNZ  CNT10,      $WAIT_10MS  ; 1[ms] x 10[count] = 10[ms]

      CLR1  PIF1              ; Clear the INTP1 interrupt request

;-----  Identification of chattering detection  -----
      BT    P4.3, $END_INTP1  ; Branch if there is no switch input

SW_ON:
;-----  Reverse LED1  -----
      XOR   P2,   #00000001B  ; Reverse output of LED1

      MOV   CNT60,      #60   ; Assign the 60 ms wait count value
;-----  60 ms wait  -----
WAIT_60MS:
      CALL  !WAIT_1MS         ; Subroutine call for a 1 ms wait
      DBNZ  CNT60,      $WAIT_60MS  ; 1[ms] x 60[count] = 60[ms]

;-----  Clear the watchdog timer  -----
      MOV   WDTE, #0ACH       ; Clear the watchdog timer

;-----  Identification of switch input status  -----
      BF    P4.3, $SW_ON      ; Branch if the switch input continues

END_INTP1:
      POP   AX                ; Restore the AX register data
```

```
      RETI                    ; Return from interrupt servicing

;------------------------------------------------------------------------------
;     Subroutine for a 1 ms wait (for 4 MHz operation)
;
;This is a subroutine for performing a rendezvous in ms units by using
;software.
;This subroutine is called by the number of times required when performing a
;rendezvous in ms units.
;The time in ms units that is indicated with #num can be measured more
;accurately by setting as follows.  (saddr: short direct addressable memory
;area)
;     MOV   saddr,      #num
;loopxx:
;     CALL  !WAIT_1MS
;     DBNZ  saddr,      $loopxx
;------------------------------------------------------------------------------
WAIT_1MS:
      PUSH  BC                ; Save the BC register data to the stack
      MOV   B,    #220        ; Assign the 220-time loop count value
LOOP_220:
      NOP
      NOP
      NOP
      NOP
      NOP
      NOP
      DBNZ  B,    $LOOP_220   ; 220-time loop for a 1 ms wait
      NOP
      NOP
      POP   BC                ; Restore the BC register data
      RET                     ; Return from the subroutine

end
```

● main.c (C language version)

```
/******************************************************************************

      NEC Electronics     78K0S/KB1+

*******************************************************************************
      78K0S/KB1+  Sample program
*******************************************************************************
      Watchdog timer
*******************************************************************************
<<History>>
      2007.6.--   Release
*******************************************************************************

<<Overview>>

This sample program presents an example of using the watchdog timer function.
The overflow time of the watchdog timer is set to 131.07 ms by counting the 8
MHz fx and runaway detection is performed.  (fx: system clock source)
After completion of the initial settings, LED1 blinks every 120 ms and blinks
every 60 ms when the switch is pressed.
Here, a reset is generated by the watchdog timer at a 50% chance of occurrence,
depending on the timing at which SW is pressed, and LED2 lights, because the
watchdog timer is cleared about 70 ms after the interrupt of the SW input was
generated.
# Normally, an overflow should be avoided by clearing the watchdog timer at
  the beginning and end of an interrupt.


  <Principal setting contents>

  - Declare a function run by an interrupt: INTP1 -> fn_intp1()
  - Select fx as the watchdog timer (WDT) count clock and
    set the overflow time to 131.07 ms
  - Set the low-voltage detection voltage (VLVI) to 2.85 V +-0.15 V
  - Generate an internal reset signal (low-voltage detector) when VDD < VLVI
after VDD >= VLVI
  - Set the CPU clock frequency to 4 MHz
  - Clear the watchdog timer about every 120 ms in the main loop
  - Clear the watchdog timer about 70 ms after the interrupt of the switch
input was generated and clear the watchdog timer about every 60 ms if the
switch input continues thereafter
  - Set the valid edge of external interrupt INTP1 to falling edge
  - Set the chattering removal time during switch input to 10 ms


  <SW input and LED1>

    +-----------------------------+
    |  SW   |        LED1         |
    | (P43) |       (P20)         |
    |-------|---------------------|
    |  OFF  | Blink every 120 ms  |
    |-------|---------------------|
    |  ON   | Blink every 60 ms   |#
    +-----------------------------+
   # A reset is generated by WDT at a 50% chance of occurrence when SW is
turned on.
```

```
     In that case, LED1 blinks every 120 ms even if SW is turned on.


  <RESET source and LED2>

    +------------------------+
    |  RESET Source  |  LED2 |
    |                | (P21) |
    |----------------|-------|
    | Other than WDT |  OFF  |
    |----------------|-------|
    | WDT            |  ON   |
    +------------------------+


<<I/O port settings>>

  Input: P43
  Output: P00-P03, P20-P23, P30-P33, P40-P42, P44-P47, P120-P123, P130
  # All unused ports are set as the output mode.

*****************************************************************************/


/*============================================================================

      Preprocessing directive (#pragma)

============================================================================*/
#pragma     SFR                     /* SFR names can be described at the C
source level */
#pragma     EI                      /* EI instructions can be described at the
C source level */
#pragma     NOP                     /* NOP instructions can be described at
the C source level */
#pragma interrupt INTP1 fn_intp1    /* Interrupt function declaration:INTP1 */

/*****************************************************************************

      Initialization after RESET

*****************************************************************************/
      sreg unsigned char ucRESF;    /* 8-bit variable for reset source check
(high-speed internal RAM area) */

void hdwinit(void){
      unsigned char ucCnt200us;     /* 8-bit variable for 200 us wait */

/*----------------------------------------------------------------------------
      Detect low-voltage + initialize the watchdog timer + set the clock
----------------------------------------------------------------------------*/
      /* Initialize the watchdog timer */
      WDTM  = 0b01101111;           /* WDT overflow time = 2^20/fx = 131.07 ms
*/

      /* Set the clock <1> */
      PCC   = 0b00000000;           /* The clock supplied to the CPU (fcpu) =
fxp (= fx/4 = 2 MHz) */
```

```
      LSRCM = 0b00000001;           /* Stop the oscillation of the low-speed
internal oscillator */

      /* Initialize the port 2 */
      P2    = 0b00000011;           /* Set output latches of P20, P21 as high
(turn off LED1, LED2) and P22, P23 as low */
      PM2   = 0b11110000;           /* Set P20-P23 as output mode */

      /* Check the reset source */
      ucRESF = RESF;          /* Read the reset source */

      if (ucRESF.4){    /* A reset generated by WDT */
            P2 = 0b00000001;        /* Light LED2 */
      }

      if (!ucRESF.0){   /* Omit subsequent LVI-related processing during LVI
reset */

            /* Set low-voltage detection */
            LVIS  = 0b00000111;     /* Set the low-voltage detection level
(VLVI) to 2.85 V +-0.15 V */
            LVION = 1;        /* Enable the low-voltage detector operation */

            for (ucCnt200us = 0; ucCnt200us < 9; ucCnt200us++){   /* Wait of
about 200 us */
                  NOP();
            }

            while (LVIF){           /* Wait for VDD >= VLVI */
                  WDTE = 0xAC;      /* Clear the watchdog timer */
            }

            LVIMD = 1;              /* Set so that an internal reset signal is
generated when VDD < VLVI */
      }

      /* Set the clock <2> */
      PPCC  = 0b00000001;           /* The clock supplied to the peripheral
hardware (fxp) = fx/2 (= 4 MHz) */
                                    /* -> The clock supplied to the CPU (fcpu)
= fxp = 4 MHz */

/*-----------------------------------------------------------------------------
      Initialize the port 0
-----------------------------------------------------------------------------*/
      P0    = 0b00000000;           /* Set output latches of P00-P03 as low */
      PM0   = 0b11110000;           /* Set P00-P03 as output mode */

/*-----------------------------------------------------------------------------
      Initialize the port 3
-----------------------------------------------------------------------------*/
      P3    = 0b00000000;           /* Set output latches of P30-P33 as low */
      PM3   = 0b11110000;           /* Set P30-P33 as output mode */

/*-----------------------------------------------------------------------------
      Initialize the port 4
-----------------------------------------------------------------------------*/
      P4    = 0b00000000;           /* Set output latches of P40-P47 as low */
```

```
      PU4   = 0b00001000;           /* Connect on-chip pull-up resistor to P43
*/
      PM4   = 0b00001000;           /* Set P43 as input mode, P40-P42 and P44-
P47 as output mode */

/*----------------------------------------------------------------------------
      Initialize the port 12
----------------------------------------------------------------------------*/
      P12   = 0b00000000;           /* Set output latches of P120-P123 as low
*/
      PM12  = 0b11110000;           /* Set P120-P123 as output mode */

/*----------------------------------------------------------------------------
      Initialize the port 13
----------------------------------------------------------------------------*/
      P13   = 0b00000001;           /* Set output latch of P130 as high */

/*----------------------------------------------------------------------------
      Set the interrupt
----------------------------------------------------------------------------*/
      INTM0 = 0b00000000;           /* Set the valid edge of INTP1 to falling
edge */
      PIF1  = 0;                    /* Clear invalid interrupt requests in
advance */
      PMK1  = 0;                    /* Release the INTP1 interrupt mask */

      return;
}

/*****************************************************************************

      Main loop

*****************************************************************************/
void main(void){
      unsigned int unCnt120ms;      /* 16-bit variable for 120 ms wait */

      EI();                         /* Enable vector interrupt */

      while (1){
            P2 ^= 0b00000001;       /* Reverse output of LED1 */

            for (unCnt120ms = 0; unCnt120ms < 6666; unCnt120ms++){      /*
Wait of about 120 ms */
                  NOP();
            }

            WDTE  = 0xAC;           /* Clear the watchdog timer */
      }
}

/*****************************************************************************

      External interrupt INTP1

*****************************************************************************/
__interrupt void fn_intp1(){
      unsigned int unCnt;     /* 16-bit variable for a wait */
```

```
      for (unCnt = 0; unCnt < 555; unCnt++){    /* Wait of about 10 ms (for
chattering removal) */
            NOP();
      }

      PIF1  = 0;                    /* Clear the INTP1 interrupt request */

      while (!P4.3){          /* Processing during SW input */
            P2 ^= 0b00000001;       /* Reverse output of LED1 */

            for (unCnt = 0; unCnt < 3333; unCnt++){   /* Wait of about 60 ms
*/
                  NOP();
            }

            WDTE  = 0xAC;           /* Clear the watchdog timer */
      }

      return;
}
```

- op.asm (Common to assembly language and C language versions)

```
;==============================================================================
;
;      Option byte
;
;==============================================================================
OPBT  CSEG  AT    0080H
        DB      10011100B       ; Option byte area
;                   ||||
;                   |||+----------- Low-speed internal oscillator can be
stopped by software
;                   |++------------ High-speed internal oscillation clock (8
MHz) is selected for system clock source
;                   +-------------- P34/RESET pin is used as RESET pin

        DB      11111111B       ; Protect byte area (for the self programming
mode)
;               ||||||||
;               ++++++++----------- All blocks can be written or erased

end
```

APPENDIX B REVISION HISTORY

The mark “<R>” shows major revised points. The revised points can be easily searched by copying an “<R>” in the PDF file and specifying it in the “Find what.” field.

Edition	Date Published	Page	Revision
1st edition	October 2007	–	–
2nd edition	September 2008	pp.19 to 21	Modification of 5.1 Building the Sample Program
		p.24	CHAPTER 6 RELATED DOCUMENTS • Addition of Flash Programming Manual (Basic) MINICUBE2 version

For further information, please contact:

NEC Electronics Corporation
1753, Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa 211-8668,
Japan
Tel: 044-435-5111
http://www.necel.com/

[America]

NEC Electronics America, Inc.
2880 Scott Blvd.
Santa Clara, CA 95050-2554, U.S.A.
Tel: 408-588-6000
800-366-9782
http://www.am.necel.com/

[Europe]

NEC Electronics (Europe) GmbH
Arcadiastrasse 10
40472 Düsseldorf, Germany
Tel: 0211-65030
http://www.eu.necel.com/

Hanover Office
Podbielskistrasse 166 B
30177 Hannover
Tel: 0 511 33 40 2-0

Munich Office
Werner-Eckert-Strasse 9
81829 München
Tel: 0 89 92 10 03-0

Stuttgart Office
Industriestrasse 3
70565 Stuttgart
Tel: 0 711 99 01 0-0

United Kingdom Branch
Cygnus House, Sunrise Parkway
Linford Wood, Milton Keynes
MK14 6NP, U.K.
Tel: 01908-691-133

Succursale Française
9, rue Paul Dautier, B.P. 52
78142 Velizy-Villacoublay Cédex
France
Tel: 01-3067-5800

Sucursal en España
Juan Esplandiu, 15
28007 Madrid, Spain
Tel: 091-504-2787

Tyskland Filial
Täby Centrum
Entrance S (7th floor)
18322 Täby, Sweden
Tel: 08 638 72 00

Filiale Italiana
Via Fabio Filzi, 25/A
20124 Milano, Italy
Tel: 02-667541

Branch The Netherlands
Steijgerweg 6
5616 HS Eindhoven
The Netherlands
Tel: 040 265 40 10

[Asia & Oceania]

NEC Electronics (China) Co., Ltd
7th Floor, Quantum Plaza, No. 27 ZhiChunLu Haidian
District, Beijing 100083, P.R.China
Tel: 010-8235-1155
http://www.cn.necel.com/

Shanghai Branch
Room 2509-2510, Bank of China Tower,
200 Yincheng Road Central,
Pudong New Area, Shanghai, P.R.China P.C:200120
Tel:021-5888-5400
http://www.cn.necel.com/

Shenzhen Branch
Unit 01, 39/F, Excellence Times Square Building,
No. 4068 Yi Tian Road, Futian District, Shenzhen,
P.R.China P.C:518048
Tel:0755-8282-9800
http://www.cn.necel.com/

NEC Electronics Hong Kong Ltd.
Unit 1601-1613, 16/F., Tower 2, Grand Century Place,
193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: 2886-9318
http://www.hk.necel.com/

NEC Electronics Taiwan Ltd.
7F, No. 363 Fu Shing North Road
Taipei, Taiwan, R. O. C.
Tel: 02-8175-9600
http://www.tw.necel.com/

NEC Electronics Singapore Pte. Ltd.
238A Thomson Road,
#12-08 Novena Square,
Singapore 307684
Tel: 6253-8311
http://www.sg.necel.com/

NEC Electronics Korea Ltd.
11F., Samik Lavied'or Bldg., 720-2,
Yeoksam-Dong, Kangnam-Ku,
Seoul, 135-080, Korea
Tel: 02-558-3737
http://www.kr.necel.com/

G0706