To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

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

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

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

78K0/Kx2-L

Sample Program (Real-Time Counter)

Fixed-Interval Interrupts and Alarm Interrupts

This document describes an operation overview of the sample program and how to use it, as well as how to set up and use the real-time counter. In the sample program, LEDs (LED1 and LED2) are controlled by using the real-time counter to generate fixed-interval interrupts and alarm interrupts.

Target device
78K0/KC2-L microcontroller

CONTENTS

Document No. U19692EJ1V0AN00 (1st edition)
Date Published September 2009 N

- **The information in this document is current as of May, 2009. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets, 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 and safety of NEC Electronics products, customers agree and acknowledge that the possibility of defects thereof cannot be eliminated entirely. In addition, NEC Electronics products are not taken measures to prevent radioactive rays in the product design. When customers use NEC Electronics products with their products, customers shall, on their own responsibility, incorporate sufficient safety measures such as redundancy, fire-containment and anti-failure features to their products in order to avoid risks of the damages to property (including public or social property) or injury (including death) to persons, as the result of defects of NEC Electronics products.
- 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).

M8E0904E

CHAPTER 1 OVERVIEW

This sample program shows an example of using the real-time counter.

In the sample program, LEDs (LED1 and LED2) are controlled by using the real-time counter to generate fixed-interval interrupts and alarm interrupts.

(1) Primary initial settings

<Option byte settings>

- Allowing the internal low-speed oscillator to be programmed to stop
- Disabling the watchdog timer
- Setting the internal high-speed oscillation clock frequency to 8 MHz
- Disabling LVI from being started by default

<Settings during initialization immediately after a reset ends>

- Specifying the ROM and RAM sizes
- Setting up I/O ports
  - Specifying P00 and P01 to be used to control the LEDs (LED1 and LED2)
- Checking whether V_{DD} is 2.7 V or more by using the low-voltage detector[Note 1]
- Specifying that the CPU clock runs on the internal high-speed oscillation clock (8 MHz)
- Specifying the XT1 oscillation mode for the subsystem clock pin
- Stopping the internal low-speed oscillator
- Disabling peripheral hardware not to be used
- Setting up the real-time counter
  - Making the system wait until the subsystem clock stabilizes by using 8-bit timer H0[Note 2] (about 1 second)[Note 3]
  - Specifying 8:59:50 a.m. as the count start time and 9:00:00 a.m. as the alarm time
  - Setting fixed-interval interrupts (generated at 0.5 second intervals) and alarm interrupts
- Enabling interrupts

Notes 1. For details about the low-voltage detector, refer to the 78K0/Kx2-L User's Manual.

2. For details about 8-bit timer H0, refer to the 78K0/Kx2-L User's Manual.

3. Adjust the oscillation stabilization time for the subsystem clock according to the resonator used.

(2) Processing after main loop

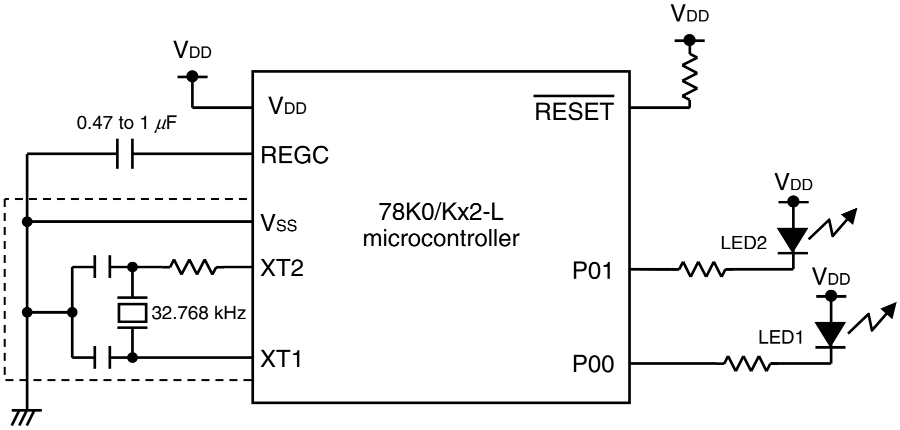
After specifying the initial settings, control the LEDs (LED1 and LED2) by using the real-time counter to generate fixed-interval interrupts and alarm interrupts. LED1 blinks every second based on interrupts generated at fixed intervals. LED2 turns on 10 seconds after the initial settings have been specified based on the generated alarm interrupt.

CHAPTER 2 CIRCUIT DIAGRAM

This chapter provides a circuit diagram and describes the devices used in this sample program other than the microcontroller.

2.1 Circuit Diagram

A circuit diagram is shown below.



Cautions 1. Use the microcontroller at a voltage in the range of 2.94 V ≤ V_{DD} ≤ 5.5 V.

2. **Connect the AV_{REF} pin directly to V_{DD}.**
3. **Connect the AV_{SS} pin directly to GND.**
4. **Connect REGC to V_{SS} via a capacitor (0.47 to 1 μF).**
5. **Handle unused pins that are not shown in the circuit diagram as follows:**
   - **I/O ports: Set them to output mode and leave them open (unconnected).**
   - **Input ports: Connect them independently to V_{DD} or V_{SS} via a resistor.**
6. **When using the XT1 oscillator, wire as follows in the area enclosed by the broken lines in the above figure to avoid an adverse effect from wiring capacitance.**
   - **Keep the wiring length as short as possible.**
   - **Do not cross the wiring with the other signal lines.**
   - **Do not route the wiring near a signal line through which a high fluctuating current flows.**
   - **Always make the ground point of the oscillator capacitor the same potential as V_{SS}.**
   - **Do not ground the capacitor to a ground pattern through which a high current flows.**
   - **Do not fetch signals from the oscillator.**
7. **The XT1 oscillator is designed as a low-amplitude circuit for reducing power consumption, and is more prone to malfunction due to noise than the X1 oscillator. Particular care is therefore required with the wiring method when the XT1 clock is used.**
8. **In this sample program, the P121/X1/TOOLC0 and P122/X2/EXCLK/TOOLD0 pins are used for on-chip debugging.**

Remark For the resonator selection and oscillator constant, customers are requested to either evaluate the oscillation themselves or apply to the resonator manufacturer for evaluation.

2.2 Used Device Other than Microcontroller

The following device is used in addition to the microcontroller:

(1) LEDs (LED1 and LED2)

LEDs receive output signals and support fixed-interval and alarm interrupts.

CHAPTER 3 SOFTWARE

This chapter describes the files included in the compressed file to be downloaded, internal peripheral functions of the microcontroller to be used, and initial settings and provides an operation overview of the sample program and the flow charts.

3.1 Included Files

The following table shows the files included in the compressed file to be downloaded.

File Name	Description	Compressed (*.zip) File Included	
		ZIP	PM32
main.asm (Assembly language version) main.c (C language version)	Source file for hardware initialization processing and main processing of microcontroller	●[Note]	●[Note]
op.asm	Assembler source file for setting the option byte (This file is used for setting up the watchdog timer and internal low-speed oscillator and selecting the internal high-speed oscillation clock frequency.)	●	●
Kx2-L_RTC.prw	Work space file for integrated development environment PM+		●
Kx2-L_RTC.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 file is included.

PM32 : 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.

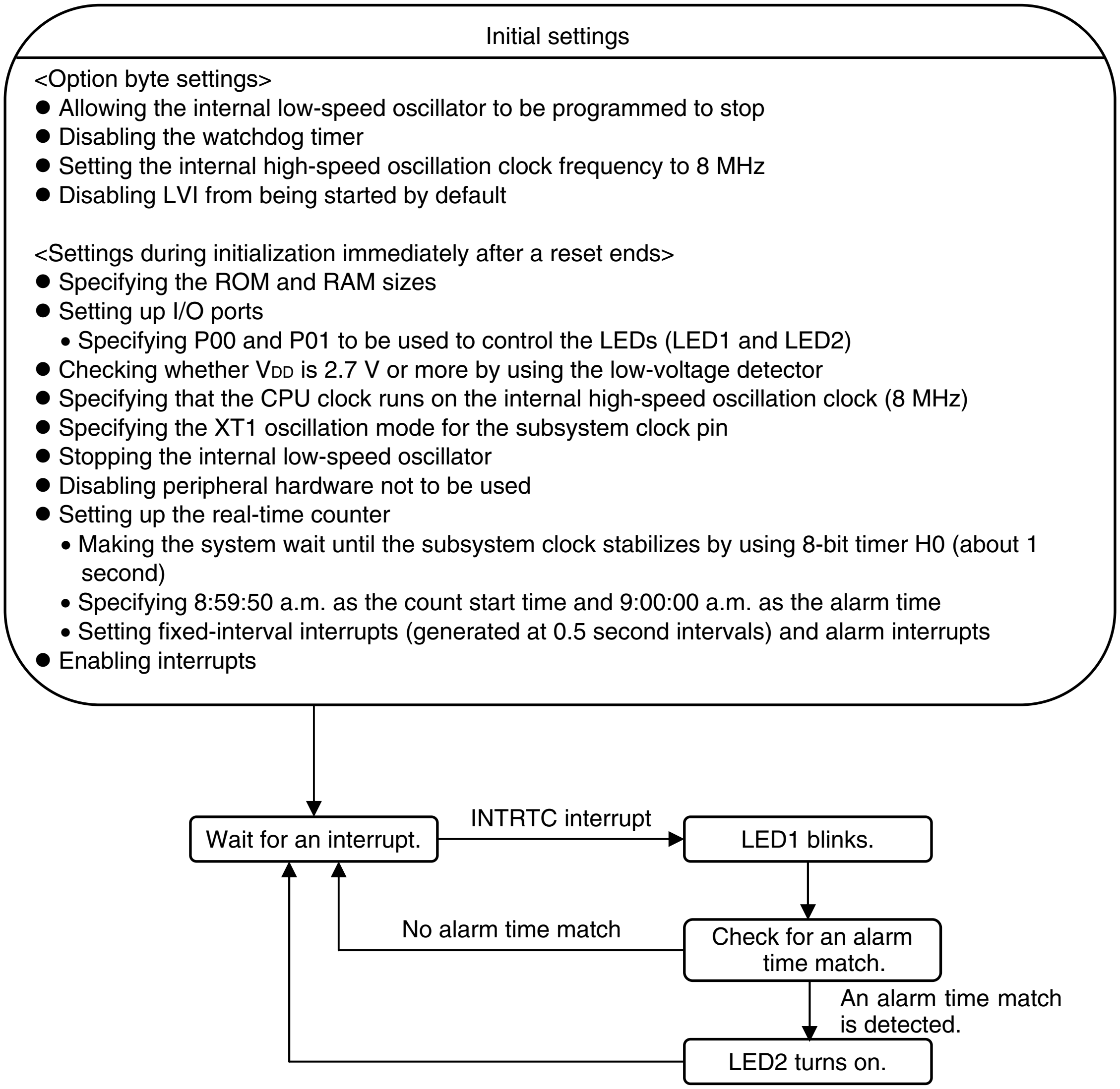
• Real-time counter:	Used to generate fixed-interval interrupts and alarm interrupts.
• P00 and P01:	Used to control the LEDs (LED1 and LED2).
• Low-voltage detector:	Checks whether V_{DD} is 2.7 V or more.
• 8-bit timer H0:	Used to make the system wait until the subsystem clock stabilizes.

3.3 Initial Settings and Operation Overview

In this sample program, initial settings including the selection of the clock frequency and setting of the I/O ports, interrupts, and real-time counter are performed.

After specifying the initial settings, control the LEDs (LED1 and LED2) by using the real-time counter to generate fixed-interval interrupts and alarm interrupts. LED1 blinks every second based on interrupts generated at fixed intervals. LED2 turns on 10 seconds after the initial settings have been specified based on the generated alarm interrupt.

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



Initial settings

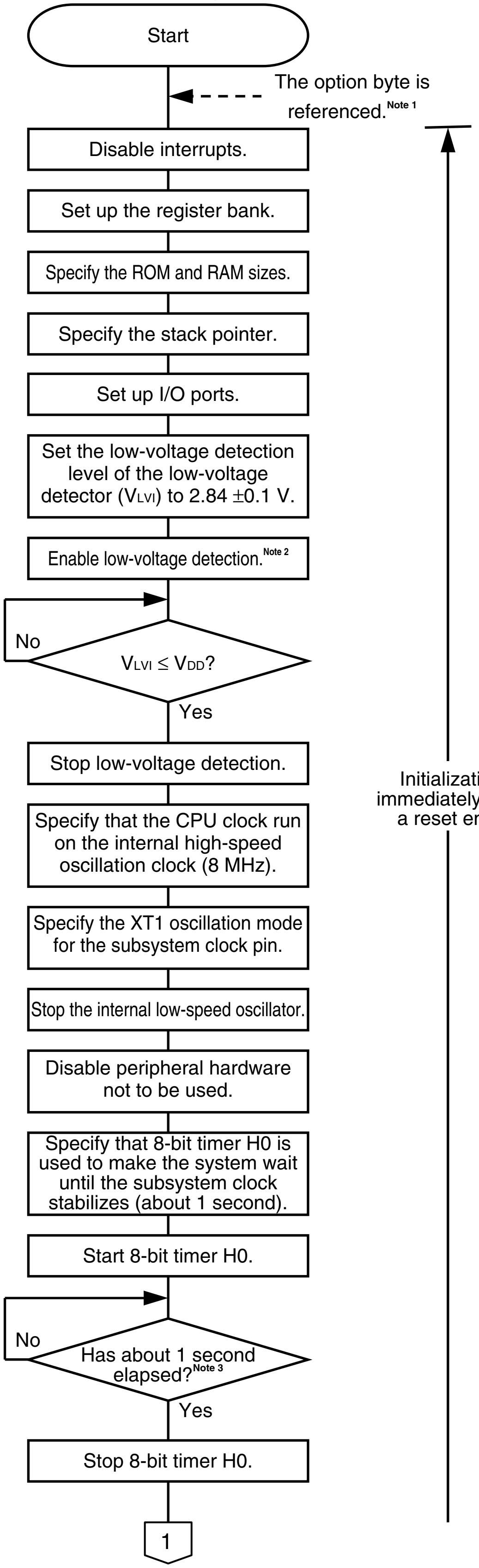
<Option byte settings>
- Allowing the internal low-speed oscillator to be programmed to stop
- Disabling the watchdog timer
- Setting the internal high-speed oscillation clock frequency to 8 MHz
- Disabling LVI from being started by default

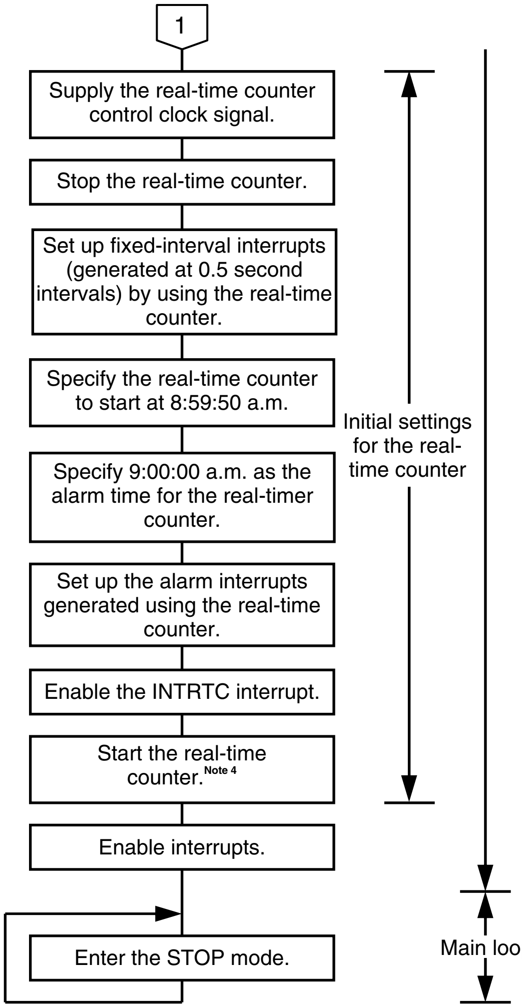
<Settings during initialization immediately after a reset ends>
- Specifying the ROM and RAM sizes
- Setting up I/O ports
  - Specifying P00 and P01 to be used to control the LEDs (LED1 and LED2)
- Checking whether V_{DD} is 2.7 V or more by using the low-voltage detector
- Specifying that the CPU clock runs on the internal high-speed oscillation clock (8 MHz)
- Specifying the XT1 oscillation mode for the subsystem clock pin
- Stopping the internal low-speed oscillator
- Disabling peripheral hardware not to be used
- Setting up the real-time counter
  - Making the system wait until the subsystem clock stabilizes by using 8-bit timer H0 (about 1 second)
  - Specifying 8:59:50 a.m. as the count start time and 9:00:00 a.m. as the alarm time
  - Setting fixed-interval interrupts (generated at 0.5 second intervals) and alarm interrupts
- Enabling interrupts

3.4 Flow Charts

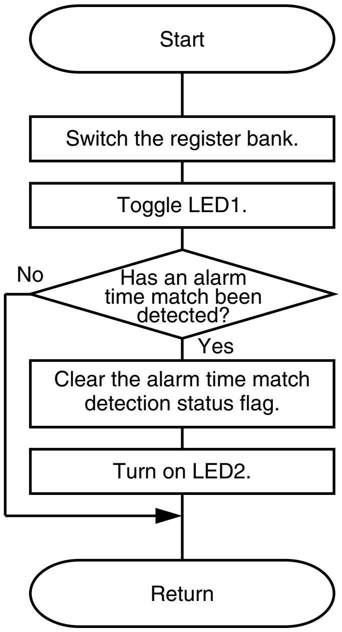
The flow charts for the sample program are shown below.

<Initialization immediately after a reset ends>





<INTRTC interrupt servicing>



Notes 1. The option byte is automatically referenced by the microcontroller immediately after a reset ends. In this sample program, the following settings are specified using the option byte:

- Allowing the internal low-speed oscillator to be programmed to stop
- Disabling the watchdog timer
- Setting the internal high-speed oscillation clock frequency to 8 MHz
- Disabling LVI from being started by default

2. The low-voltage detector is enabled, and then the system is made to wait at least 10 μs until the low-voltage detector stabilizes.

3. Adjust the oscillation stabilization time for the subsystem clock according to the resonator used.

4. If the STOP mode is entered before the real-time counter stabilizes, the real-time counter might be disabled. Therefore, in this sample program, the system is made to wait for two cycles of the subsystem clock (about 62 μs) before starting the real-time counter.

CHAPTER 4 SETTING METHODS

This chapter describes how to set up the real-time counter and adjust the time, and provides a software coding example.

For other initial settings, refer to the 78K0/Kx2-L Sample Program (Initial Settings) LED Lighting Switch Control Application Note.

For how to set registers, refer to the 78K0/Kx2-L User's Manual.

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

4.1 Setting up Real-Time Counter

The real-time counter is controlled by the following 13 registers:

- Peripheral enable register 0 (PER0)
- Real-time counter control register 0 (RTCC0)
- Real-time counter control register 1 (RTCC1)
- Second count register (SEC)
- Minute count register (MIN)
- Hour count register (HOUR)
- Day count register (DAY)
- Week count register (WEEK)
- Month count register (MONTH)
- Year count register (YEAR)
- Alarm minute register (ALARMWM)
- Alarm hour register (ALARMWH)
- Alarm week register (ALARMWW)

[Example of the setup procedure when using fixed-interval interrupts (generated at 0.5 second intervals) and alarm interrupts generated using the real-time counter]

(The same procedure is used in the sample program.)

<1> Set bit 7 (RTCEN) of PER0 to 1 (to supply the real-time counter control clock signal).^{Note 1}
<2> Clear bit 7 (RTCE) of RTCC0 to 0 (to stop the real-time counter).
<3> Use bit 3 (AMPM) of RTCC0 to select between the 12-hour system and 24-hour system.
<4> Use bits 2 to 0 (CT2 to CT0) of RTCC0 to select the interval at which to generate a fixed-interval interrupt (INTRTC).
<5> Specify the count start time for SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR.
<6> Clear bit 7 (WALE) of RTCC1 to 0 (to disable alarm time matching).
<7> Set bit 6 (WALIE) of RTCC1 to 1 (to generate an interrupt upon an alarm time match).
<8> Specify the alarm time for ALARMWM, ALARMWH, and ALARMWW.
<9> Set bit 7 (WALE) of RTCC1 to 1 (to enable alarm time matching).
<10> Clear the INTRTC interrupt request (RTCIF = 0).
<11> Enable the INTRTC interrupt (RTCMK = 0).
<12> Set bit 7 (RTCE) of RTCC0 to 1 (to enable the real-time counter).^{Note 2}

Notes 1. Set RTCEN to 1 while the subsystem clock (f_{SUB}) is stable.

2. When entering the STOP mode immediately after setting RTCE to 1, first wait for two cycles of the subsystem clock (f_{SUB}) (about 62 μs) or more.

Remarks 1. ALARMWM, ALARMWH, and ALARMWW may be written in any order.

2. Fixed-interval interrupts and alarm match interrupts use the same interrupt source (INTRTC). When using these two types of interrupts at the same time, which interrupt occurred can be judged by checking the fixed-interval interrupt status flag (RIFG) and the alarm detection status flag (WAFG) upon INTRTC occurrence.

(1) Peripheral enable register 0 (PER0)

This register controls the supply of the real-time counter control clock.

Figure 4-1. Format of Peripheral Enable Register 0 (PER0)

RTCEN	**0**	**0**	**0**	**0**	**0**	**0**	**0**

RTCEN	Control of real-time counter (RTC) control clock[Note]
0	Stops supply of control clock. • SFR used by the real-time counter cannot be written (can be read). • Operation of the real-time counter continues.
1	Supplies control clock. • SFR used by the real-time counter can be read and written.

Note The control clock supply stopped by clearing RTCEN to 0 is the clock to be used when write-accessing the registers (such as the RTCC0 register) to be used for the real-time counter (RTC) from the CPU. The RTC operating clock (f_{SUB}) does not stop, even if RTCEN is cleared to 0.

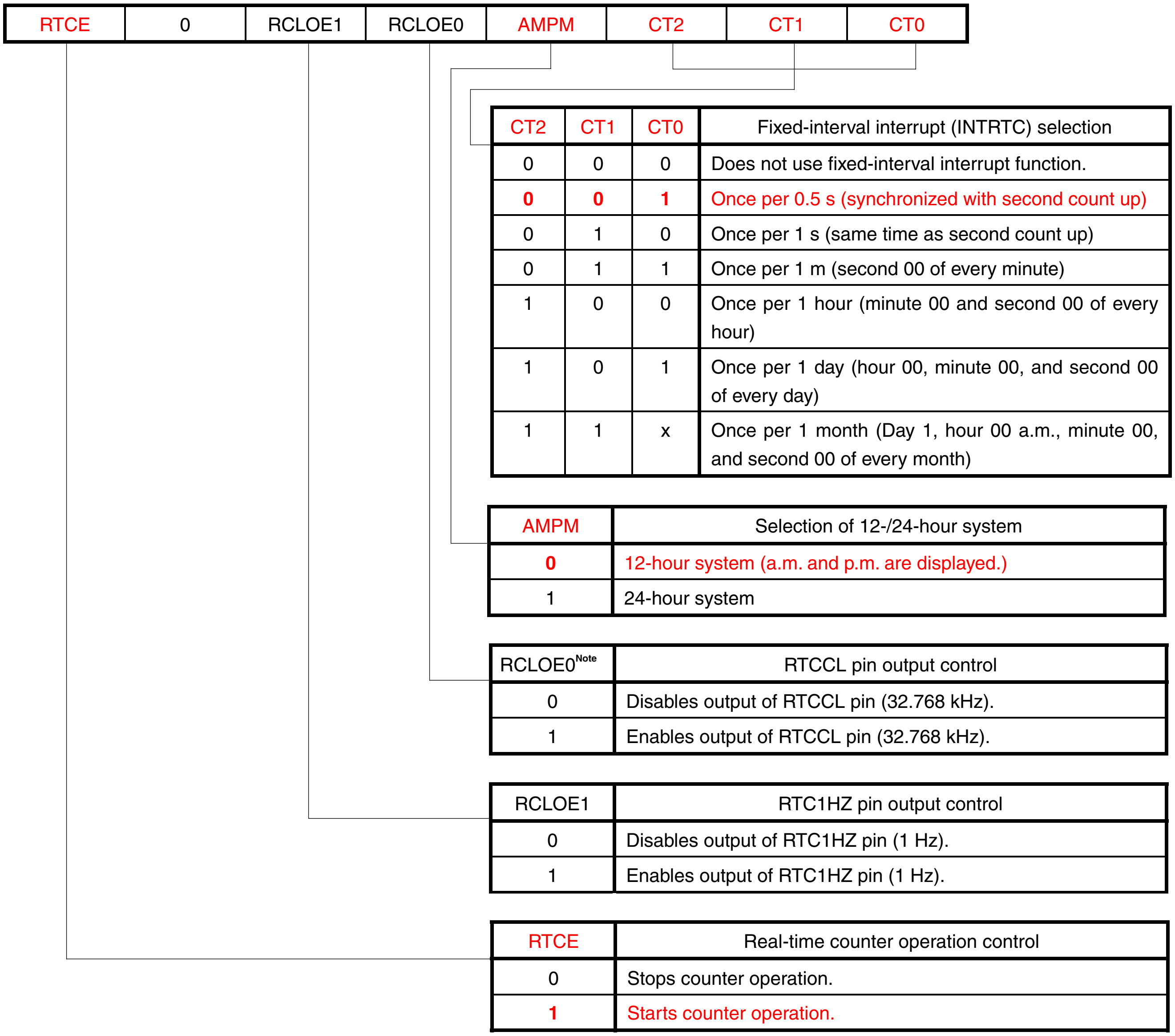
Caution Be sure to clear bits 6 to 0 to “0”.

Remark The values written in red in the above figure are specified in this sample program.

(2) Real-time counter control register 0 (RTCC0)

The RTCC0 register is an 8-bit register that is used to start or stop the real-time counter operation, control the RTCCL and RTC1HZ pins, and set a 12- or 24-hour system and the fixed-interval interrupt function.

Figure 4-2. Format of Real-Time Counter Control Register 0 (RTCC0)



RTCE	0	RCLOE1	RCLOE0	AMPM	CT2	CT1	CT0

CT2	CT1	CT0	Fixed-interval interrupt (INTRTC) selection
0	0	0	Does not use fixed-interval interrupt function.
0	**0**	**1**	Once per 0.5 s (synchronized with second count up)
0	1	0	Once per 1 s (same time as second count up)
0	1	1	Once per 1 m (second 00 of every minute)
1	0	0	Once per 1 hour (minute 00 and second 00 of every hour)
1	0	1	Once per 1 day (hour 00, minute 00, and second 00 of every day)
1	1	x	Once per 1 month (Day 1, hour 00 a.m., minute 00, and second 00 of every month)

AMPM	Selection of 12-/24-hour system
0	12-hour system (a.m. and p.m. are displayed.)
1	24-hour system

RCLOE0[Note]	RTCCL pin output control
0	Disables output of RTCCL pin (32.768 kHz).
1	Enables output of RTCCL pin (32.768 kHz).

RCLOE1	RTC1HZ pin output control
0	Disables output of RTC1HZ pin (1 Hz).
1	Enables output of RTC1HZ pin (1 Hz).

RTCE	Real-time counter operation control
0	Stops counter operation.
1	Starts counter operation.

Note RCLOE0 and RCLOE2 of real-time counter control register 2 (RTCC2) must not be enabled at the same time.

Cautions 1. To change the value of AMPM, set RWAIT (bit 0 of RTCC1) to 1, and re-set the hour count register (HOUR).

2. After changing the values of CT2 to CT0, clear the interrupt request flag.

3. Be sure to clear bit 6 to “0”.

Remarks 1. x: don’t care

2. The values written in red in the above figure are specified in this sample program.

(3) Real-time counter control register 1 (RTCC1)

The RTCC1 register is an 8-bit register that is used to control the alarm interrupt function and the wait time of the counter.

Figure 4-3. Format of Real-Time Counter Control Register 1 (RTCC1)

WALE	WALIE	0	WAFG	RIFG	0	RWST	RWAIT

RWAIT	Wait control of real-time counter
0	Sets counter operation.
1	Stops SEC to YEAR counters. Mode to read or write counter value

RWST	Wait status flag of real-time counter
0	Counter is operating.
1	Mode to read or write counter value

RIFG	Fixed-interval interrupt status flag
0	Fixed-interval interrupt is not generated.
1	Fixed-interval interrupt is generated.

WAFG	Alarm detection status flag
0	Alarm mismatch
1	Detection of matching of alarm

WALIE	Control of alarm interrupt (INTRTC) function operation
0	Does not generate interrupt on matching of alarm.
1	Generates interrupt on matching of alarm.

WALE	Alarm operation control
0	Match operation is invalid.
1	Match operation is valid.

Cautions 1. To set the registers of alarm (WALIE flag of RTCC1, ALARMWM register, ALARMWH register, and ALARMWW register), disable WALE (clear it to "0").

2. WAFG is a status flag that indicates detection of matching with the alarm. It is valid only when WALE = 1 and is set to "1" one clock (32.768 kHz) after matching of the alarm is detected. This flag is cleared when "0" is written to it. Writing "1" to it is invalid.

3. RIFG indicates the status of generation of the fixed-interval interrupt. When the fixed-interval interrupt is generated, it is set to "1". This flag is cleared when "0" is written to it. Writing "1" to it is invalid.

4. RWST indicates the status whether the setting of RWAIT is valid. Before reading or writing the counter value, confirm that the value of this flag is 1.

5. WALE controls the operation of the counter. Be sure to write "1" to it to read or write the counter value. Because RSUBC continues operation, complete reading or writing of it in 1 second, and clear this bit back to 0. When RWAIT = 1, it takes up to 1 clock (32.768 kHz) until the counter value can be read or written. If RSUBC overflows when RWAIT = 1, it counts up after RWAIT = 0. If the second count register is written, however, it does not count up because RSUBC is cleared.

6. If writing is performed to the RTCC1 register with a 1-bit manipulation instruction, the RIFG flag and WAFG flag may be cleared. Therefore, to perform writing to the RTCC1 register, be sure to use an 8-bit manipulation instruction. To prevent the RIFG flag and WAFG flag from being cleared during writing, disable writing by setting 1 to the corresponding bit. If the RIFG flag and WAFG flag are not used and the value may be changed, the RTCC1 register may be written by using a 1-bit manipulation instruction.

7. Be sure to clear bits 5 and 2 to "0".

Remarks 1. Fixed-interval interrupts and alarm match interrupts use the same interrupt source (INTRTC). When using these two types of interrupts at the same time, which interrupt occurred can be judged by checking the fixed-interval interrupt status flag (RIFG) and the alarm detection status flag (WAFG) upon INTRTC occurrence.

2. The values written in red in the above figure are specified in this sample program.

(4) Second count register (SEC)

The SEC register is an 8-bit register that takes a value of 0 to 59 (decimal) and indicates the count value of seconds.

It counts up when the sub-count register (RSUBC)[Note] that counts 1 second with a clock of 32.768 kHz overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 00 to 59 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.

Figure 4-4. Format of Second Count Register (SEC)

0	SEC40	SEC20	SEC10	SEC8	SEC4	SEC2	SEC1

Note For details about the sub-count register (RSUBC), refer to the 78K0/Kx2-L User’s Manual.

Remark In this sample program, 50H is specified as the initial value.

(5) Minute count register (MIN)

The MIN register is an 8-bit register that takes a value of 0 to 59 (decimal) and indicates the count value of minutes.

It counts up when the second counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 00 to 59 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.

Figure 4-5. Format of Minute Count Register (MIN)

0	MIN40	MIN20	MIN10	MIN8	MIN4	MIN2	MIN1

Remark In this sample program, 59H is specified as the initial value.

(6) Hour count register (HOUR)

The HOUR register is an 8-bit register that takes a value of 00 to 23 or 01 to 12, 21 to 32 (decimal) and indicates the count value of hours.

It counts up when the minute counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 00 to 23, 01 to 12, or 21 to 32 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.

Figure 4-6. Format of Hour Count Register (HOUR)

0	0	HOUR20	HOUR10	HOUR8	HOUR4	HOUR2	HOUR1

Caution Bit 5 (HOUR20) of HOUR indicates AM(0)/PM(1) if AMPM = 0 (if the 12-hour system is selected). The details are shown in the table below.

24-Hour Display (AMPM Bit = 1)		12-Hour Display (AMPM Bit = 0)	
Time	HOUR Register	Time	HOUR Register
0	00H	0 a.m.	12H
1	01H	1 a.m.	01H
2	02H	2 a.m.	02H
3	03H	3 a.m.	03H
4	04H	4 a.m.	04H
5	05H	5 a.m.	05H
6	06H	6 a.m.	06H
7	07H	7 a.m.	07H
8	08H	8 a.m.	08H
9	09H	9 a.m.	09H
10	10H	10 a.m.	10H
11	11H	11 a.m.	11H
12	12H	0 p.m.	32H
13	13H	1 p.m.	21H
14	14H	2 p.m.	22H
15	15H	3 p.m.	23H
16	16H	4 p.m.	24H
17	17H	5 p.m.	25H
18	18H	6 p.m.	26H
19	19H	7 p.m.	27H
20	20H	8 p.m.	28H
21	21H	9 p.m.	29H
22	22H	10 p.m.	30H
23	23H	11 p.m.	31H

Remark In this sample program, 08H is specified as the initial value.

(7) Day count register (DAY)

The DAY register is an 8-bit register that takes a value of 1 to 31 (decimal) and indicates the count value of days.

It counts up when the hour counter overflows.

This counter counts as follows:

- 01 to 31 (January, March, May, July, August, October, December)
- 01 to 30 (April, June, September, November)
- 01 to 29 (February, leap year)
- 01 to 28 (February, normal year)

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 01 to 31 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.

Figure 4-7. Format of Day Count Register (DAY)

0	0	DAY20	DAY10	DAY8	DAY4	DAY2	DAY1

Remark In this sample program, 01H is specified as the initial value.

(8) Week count register (WEEK)

The WEEK register is an 8-bit register that takes a value of 0 to 6 (decimal) and indicates the count value of weekdays.

It counts up in synchronization with the day counter.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 00 to 06 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.

Figure 4-8. Format of Week Count Register (WEEK)

0	0	0	0	0	WEEK4	WEEK2	WEEK1

Caution **Values corresponding to the month count register and day count register are not automatically stored to the week count register. Set the week count register as follows, after reset release:**

Day	WEEK Register
Sunday	00H
Monday	01H
Tuesday	02H
Wednesday	03H
Thursday	04H
Friday	05H
Saturday	06H

Remark In this sample program, 06H is specified as the initial value.

(9) Month count register (MONTH)

The MONTH register is an 8-bit register that takes a value of 1 to 12 (decimal) and indicates the count value of months.

It counts up when the day counter overflows.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 01 to 12 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.

Figure 4-9. Format of Month Count Register (MONTH)

0	0	0	MONTH10	MONTH8	MONTH4	MONTH2	MONTH1

Remark In this sample program, 01H is specified as the initial value.

(10) Year count register (YEAR)

The YEAR register is an 8-bit register that takes a value of 0 to 99 (decimal) and indicates the count value of years.

It counts up when the month counter overflows.

Values 00, 04, 08, …, 92, and 96 indicate a leap year.

When data is written to this register, it is written to a buffer and then to the counter up to 2 clocks (32.768 kHz) later.

Set a decimal value of 00 to 99 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.

Figure 4-10. Format of Year Count Register (YEAR)

YEAR80	YEAR40	YEAR20	YEAR10	YEAR8	YEAR4	YEAR2	YEAR1

Remark In this sample program, 00H is specified as the initial value.

(11) Alarm minute register (ALARMWM)

This register is used to set minutes of alarm.

Figure 4-11. Format of Alarm Minute Register (ALARMWM)

0	WM40	WM20	WM10	WM8	WM4	WM2	WM1

Caution **Set a decimal value of 00 to 59 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.**

Remark In this sample program, 00H is specified as the initial value.

(12) Alarm hour register (ALARMWH)

This register is used to set hours of alarm.

Figure 4-12. Format of Alarm Hour Register (ALARMWH)

0	0	WH20	WH10	WH8	WH4	WH2	WH1

Cautions 1. Set a decimal value of 00 to 23, 01 to 12, or 21 to 32 to this register in BCD code. If a value outside the range is specified, the value is incremented and the counter returns to 00H when an overflow occurs.

2. Bit 5 (WH20) of ALARMWH indicates AM(0)/PM(1) if AMPM = 0 (if the 12-hour system is selected).

Remark In this sample program, 09H is specified as the initial value.

(13) Alarm week register (ALARMWW)

This register is used to set date of alarm.

Figure 4-13. Format of Alarm Week Register (ALARMWW)

0	WW6	WW5	WW4	WW3	WW2	WW1	WW0

Remarks 1. Here is an example of setting the alarm.

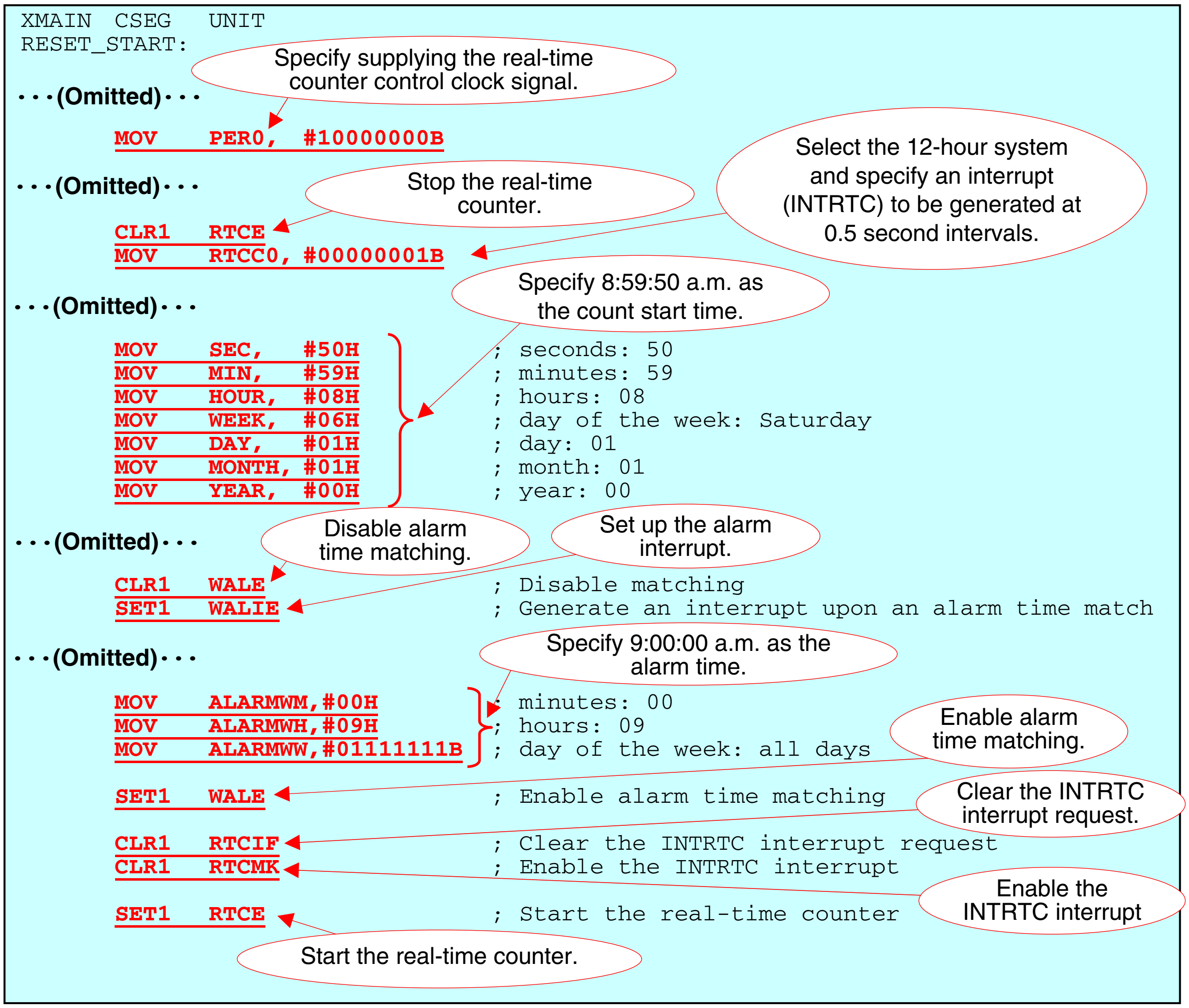
Time of Alarm	Day							12-Hour Display				24-Hour Display			
	Sunday WW0	Monday WW1	Tuesday WW2	Wednesday WW3	Thursday WW4	Friday WW5	Saturday WW6	Hour 10	Hour 1	Minute 10	Minute 1	Hour 10	Hour 1	Minute 10	Minute 1
Every day, 0:00 a.m.	1	1	1	1	1	1	1	1	2	0	0	0	0	0	0
Every day, 1:30 a.m.	1	1	1	1	1	1	1	0	1	3	0	0	1	3	0
Every day, 11:59 a.m.	1	1	1	1	1	1	1	1	1	5	9	1	1	5	9
Monday through Friday, 0:00 p.m.	0	1	1	1	1	1	0	3	2	0	0	1	2	0	0
Sunday, 1:30 p.m.	1	0	0	0	0	0	0	2	1	3	0	1	3	3	0
Monday, Wednesday, Friday, 11:59 p.m.	0	1	0	1	0	1	0	3	1	5	9	2	3	5	9

2. In this sample program, 7FH is specified as the initial value.

4.2 Software Coding Example

The settings specified for the real-time counter in the sample program are shown below as a software coding example.

(1) Assembly language



```
XMAIN  CSEG   UNIT
RESET_START:

・・・(Omitted)・・・

       MOV    PER0,  #10000000B

・・・(Omitted)・・・

       CLR1   RTCE
       MOV    RTCC0, #00000001B

・・・(Omitted)・・・

       MOV    SEC,   #50H                       ; seconds: 50
       MOV    MIN,   #59H                       ; minutes: 59
       MOV    HOUR,  #08H                       ; hours: 08
       MOV    WEEK,  #06H                       ; day of the week: Saturday
       MOV    DAY,   #01H                       ; day: 01
       MOV    MONTH, #01H                       ; month: 01
       MOV    YEAR,  #00H                       ; year: 00

・・・(Omitted)・・・

       CLR1   WALE                              ; Disable matching
       SET1   WALIE                             ; Generate an interrupt upon an alarm time match

・・・(Omitted)・・・

       MOV    ALARMWM,#00H                      ; minutes: 00
       MOV    ALARMWH,#09H                      ; hours: 09
       MOV    ALARMWW,#01111111B                ; day of the week: all days

       SET1   WALE                              ; Enable alarm time matching

       CLR1   RTCIF                             ; Clear the INTRTC interrupt request
       CLR1   RTCMK                             ; Enable the INTRTC interrupt

       SET1   RTCE                              ; Start the real-time counter
```

Specify supplying the real-time counter control clock signal.

Stop the real-time counter.

Select the 12-hour system and specify an interrupt (INTRTC) to be generated at 0.5 second intervals.

Specify 8:59:50 a.m. as the count start time.

Disable alarm time matching.

Set up the alarm interrupt.

Specify 9:00:00 a.m. as the alarm time.

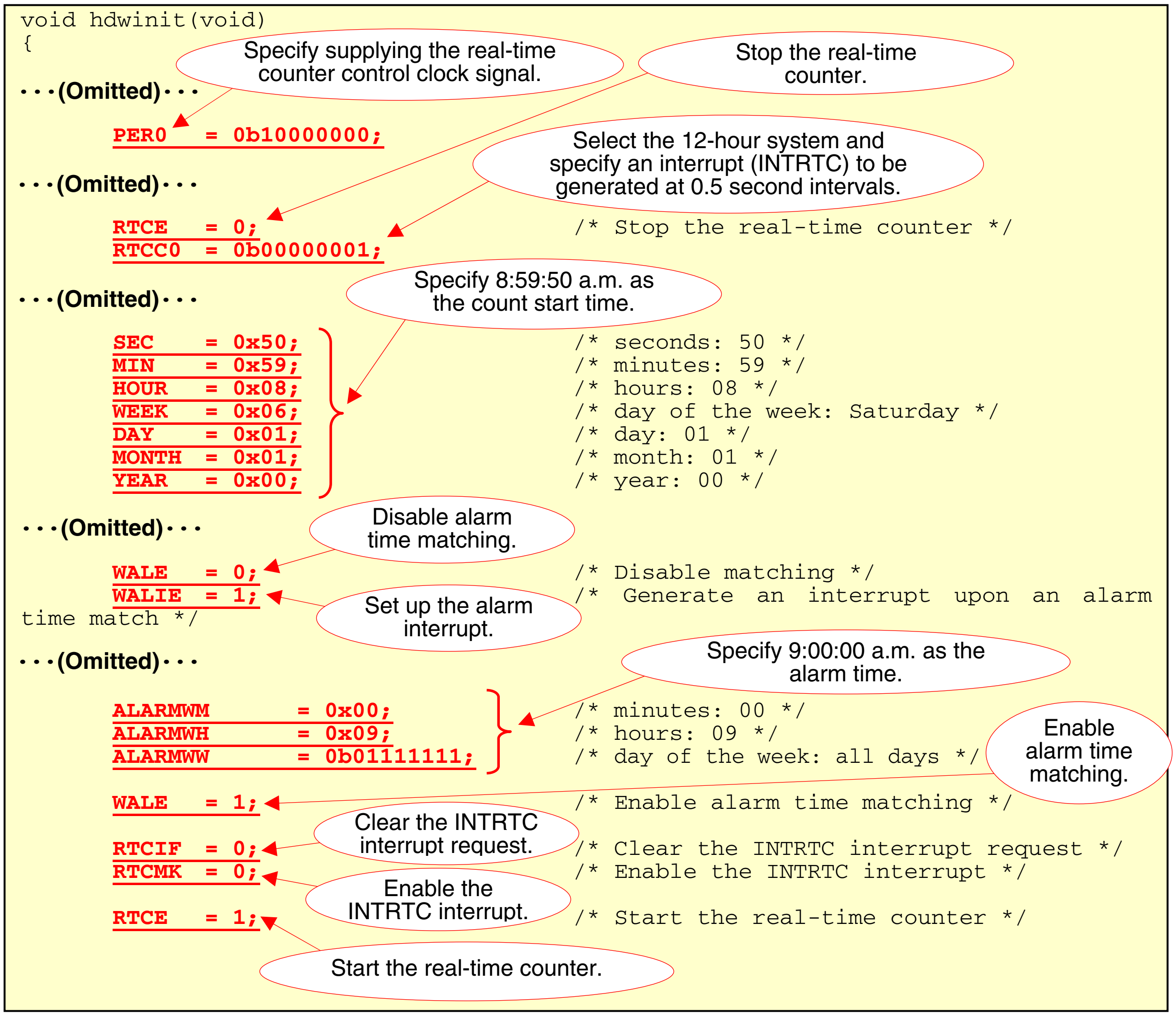
Enable alarm time matching.

Clear the INTRTC interrupt request.

Enable the INTRTC interrupt

Start the real-time counter.

(2) C language



```
void hdwinit(void)
{
・・・(Omitted)・・・

        PER0    = 0b10000000;

・・・(Omitted)・・・

        RTCE    = 0;                         /* Stop the real-time counter */
        RTCC0   = 0b00000001;

・・・(Omitted)・・・

        SEC     = 0x50;                      /* seconds: 50 */
        MIN     = 0x59;                      /* minutes: 59 */
        HOUR    = 0x08;                      /* hours: 08 */
        WEEK    = 0x06;                      /* day of the week: Saturday */
        DAY     = 0x01;                      /* day: 01 */
        MONTH   = 0x01;                      /* month: 01 */
        YEAR    = 0x00;                      /* year: 00 */

・・・(Omitted)・・・

        WALE    = 0;                         /* Disable matching */
        WALIE   = 1;                         /*  Generate  an  interrupt  upon  an  alarm
time match */

・・・(Omitted)・・・

        ALARMWM         = 0x00;              /* minutes: 00 */
        ALARMWH         = 0x09;              /* hours: 09 */
        ALARMWW         = 0b01111111;        /* day of the week: all days */

        WALE    = 1;                         /* Enable alarm time matching */

        RTCIF   = 0;                         /* Clear the INTRTC interrupt request */
        RTCMK   = 0;                         /* Enable the INTRTC interrupt */

        RTCE    = 1;                         /* Start the real-time counter */
```

4.3 Time Adjustment

This section describes how to adjust the time for the real-time counter.

(1) Watch error correction register (SUBCUD)

Use the watch error correction register (SUBCUD) to adjust the time for the real-time counter. This register is used to correct the watch with high accuracy when it is slow or fast by changing the value that overflows from the sub-count register (RSUBC) to the second count register (reference value: 7FFFH).

Figure 4-14. Format of Watch Error Correction Register (SUBCUD)

DEV	F6	F5	F4	F3	F2	F1	F0

F6	Setting of watch error correction value
0	Increases by {(F5, F4, F3, F2, F1, F0) – 1} × 2
1	Decreases by {(/F5, /F4, /F3, /F2, /F1, /F0) + 1} × 2[Note]

DEV	Setting of watch error correction timing
0	Corrects watch error when the second digits are at 00, 20, or 40 (every 20 seconds).
1	Corrects watch error only when the second digits are at 00 (every 60 seconds).

Note /F5 to /F0 are the inverted values of the corresponding bits (000011 when 111100).

Cautions 1. When (F6, F5, F4, F3, F2, F1, F0) = (*, 0, 0, 0, 0, 0, *), the watch error is not corrected. (* is 0 or 1.)

Range of correction value: (When F6 = 0) 2, 4, 6, 8, … , 120, 122, 124

(When F6 = 1) –2, –4, –6, –8, … , –120, –122, –124

2. The range of value that can be corrected by using the watch error correction register (SUBCUD) is shown below.

	DEV = 0 (Correction Every 20 Seconds)	DEV = 1 (Correction Every 60 Seconds)
Correctable range	–189.2 ppm to 189.2 ppm	–63.1 ppm to 63.1 ppm
Maximum quantization error	±1.53 ppm	±0.51 ppm
Minimum resolution	±3.05 ppm	±1.02 ppm

3. When adjusting the time, the correction values to set to F6 to F0 (2, 4, 6, …, 124, or –2, –4, –6, …, –124) can be calculated using the following equation:

(If DEV is 0)

Correction value = number of correction counts per minute ÷ 3

= (oscillation frequency ÷ target frequency – 1) × 32,768 × 60 ÷ 3

(If DEV is 1)

Correction value = number of correction counts per minute

= (oscillation frequency ÷ target frequency – 1) × 32,768 × 60

Oscillation frequency: Subsystem clock (f_{SUB}) value

Target frequency: The frequency corrected using the watch error correction register

Remark If a correctable range is –63.1 ppm or lower and 63.1 ppm or higher, set 0 to DEV.

(2) Example of adjusting the time

An example of adjusting the time for the real-time counter is shown below.

[Example of adjusting the time if the oscillation frequency is 32,772.3 Hz (32,768 Hz + 131.2 ppm)]

<1> Use the RTCCL pin or RTC1HZ pin[Note 1] to output the oscillation frequency (32,772.3 Hz) to outside the microcontroller.

<2> Measure the output oscillation frequency outside the microcontroller by using a measuring instrument and calculate the correction value based on the difference with the target frequency (32,768 Hz).[Note 2]

<3> Receive the correction values via a serial interface and set these values to the watch error correction register (SUBCUD).[Note 3]

Notes 1. For how to set the output of the RTC1HZ and RTCCL pins, refer to the 78K0/Kx2-L User's Manual.

2. In this case, the time must be adjusted every 20 seconds. Therefore, the equation for calculating the correction value is as follows:

Correction value = number of correction counts per minute ÷ 3

= (number of counts per minute at the oscillation frequency – number of counts per minute at the target frequency) ÷ 3

= (oscillation frequency ÷ target frequency × 32,768 × 60 – 32,768 × 60) ÷ 3

= (oscillation frequency ÷ target frequency – 1) × 32,768 × 60 ÷ 3

= (32,772.3 ÷ 32,768 – 1) × 32,768 × 60 ÷ 3

= 86

3. Set the following values to the watch error correction register (SUBCUD):

- Bit 7 (DEV)

  Clear DEV to 0 because the time must be adjusted every 20 seconds.

- Bits 6 to 0 (F6 to F0)

  If the correction value is 0 or more (when delaying the time), clear F6 to 0.

  Calculate F5, F4, F3, F2, F1, and F0 based on the correction value.

  {(F5, 4, F3, F2, F1, F0) – 1} × 2 = 86

  (F5, F4, F3, F2, F1, F0) = 44

  (F5, F4, F3, F2, F1, F0) = (1, 0, 1, 1, 0, 0)

CHAPTER 5 RELATED DOCUMENTS

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

Document Name			English
78K0/Kx2-L User's Manual			PDF
78K/0 Series Instructions User's Manual			PDF
RA78K0 Assembler Package User's Manual		Language	PDF
		Operation	PDF
CC78K0 C Compiler User's Manual		Language	PDF
		Operation	PDF
PM+ Project Manager User's Manual			PDF
78K0/Kx2-L Application Note	Sample Program (Initial Settings) LED Lighting Switch Control		PDF

APPENDIX A PROGRAM LIST

As a program list example, the 78K0/KC2-L microcontroller source program is shown below.

- main.asm (assembly language version)

```
  ;*******************************************************************************
  ;
  ;    NEC Electronics     78K0/KC2-L Series
  ;
  ;*******************************************************************************
  ;    78K0/KC2-L Series   Sample Program (Real-Time Counter)
  ;*******************************************************************************
  ;    Fixed-Interval Interrupts and Alarm Interrupts
  ;*******************************************************************************
  ;<<History>>
  ;    2009.1.--    Release
  ;*******************************************************************************
  ;
  ;<<Overview>>
  ;
  ; This sample program presents an example of using the real-time counter.  In the
  ; sample program, LEDs (LED1 and LED2) are controlled by using the real-time counter
  ; to generate fixed-interval interrupts and alarm interrupts.
  ;
  ;
  ; <Primary initial settings>
  ;
  ; (Option byte settings)
  ; - Allowing the internal low-speed oscillator to be programmed to stop
  ; - Disabling the watchdog timer
  ; - Setting the internal high-speed oscillation clock frequency to 8 MHz
  ; - Disabling LVI from being started by default
  ; (Settings during initialization immediately after a reset ends)
  ; - Specifying the ROM and RAM sizes
  ; - Setting up I/O ports
  ;   → Specifying P00 and P01 to be used to control the LEDs (LED1 and LED2)
  ; - Checking whether VDD is 2.7 V or more by using the low-voltage detector
  ; - Specifying that the CPU clock runs on the internal high-speed oscillation clock (8
MHz)
  ; - Specifying the XT1 oscillation mode for the subsystem clock pin
  ; - Stopping the internal low-speed oscillator
  ; - Disabling peripheral hardware not to be used
  ; - Setting up the real-time counter
  ;   → Making the system wait until the subsystem clock stabilizes by using 8-bit timer
H0 (about 1 second)
  ;   → Specifying 8:59:50 a.m. as the count start time and 9:00:00 a.m. as the alarm time
  ;   → Setting fixed-interval interrupts (generated at 0.5 second intervals) and alarm
```

```
interrupts
  ; - Enabling interrupts
  ;
  ;
  ; <LED control timing>
  ;
  ; +----------------------------------------------+
  ; | LED            | LED control timing          |
  ; |----------------------------------------------|
  ; | LED1 (P00)     | Blink every second.         |
  ; | LED2 (P01)     | Turn on at the alarm time.  |
  ; +----------------------------------------------+
  ; * The LEDs turn off if 1 is output from the ports and turn on if 0 is output from the
ports.
  ;
  ;
  ; <I/O port settings>
  ;  Output: P00, P01
  ;  * Set all unused ports that can be specified as output ports as output ports.
  ;
  ;*******************************************************************************


  ;===============================================================================
  ;
  ;    Vector table
  ;
  ;===============================================================================
  XVECT1            CSEG   AT     0000H
       DW     RESET_START         ;0000H RESET input, POC, LVI, WDT
  XVECT2            CSEG   AT     0004H
       DW     IINIT               ;0004H INTLVI
       DW     IINIT               ;0006H INTP0
       DW     IINIT               ;0008H INTP1
       DW     IINIT               ;000AH INTP2
       DW     IINIT               ;000CH INTP3
       DW     IINIT               ;000EH INTP4
       DW     IINIT               ;0010H INTP5
       DW     IINIT               ;0012H INTSRE6
       DW     IINIT               ;0014H INTSR6
       DW     IINIT               ;0016H INTST6
       DW     IINIT               ;0018H INTCSI10
       DW     IINIT               ;001AH INTTMH1
       DW     IINIT               ;001CH INTTMH0
       DW     IINIT               ;001EH INTTM50
       DW     IINIT               ;0020H INTTM000
       DW     IINIT               ;0022H INTTM010
       DW     IINIT               ;0024H INTAD
```

```
     DW     IINIT               ;0026H INTP6
     DW     IINIT               ;0028H INTRTCI
     DW     IINIT               ;002AH INTTM51
     DW     IINIT               ;002CH INTKR
     DW     IINIT               ;002EH INTRTC
     DW     IINIT               ;0030H INTP7
     DW     IINIT               ;0032H INTP8
     DW     IINIT               ;0034H INTIICA0
     DW     IINIT               ;0036H INTCSI11
     DW     IINIT               ;0038H INTP9
     DW     IINIT               ;003AH INTP10
     DW     IINIT               ;003CH INTP11
     DW     IINIT               ;003EH BRK

;===============================================================================
;
;    Define the memory stack area
;
;===============================================================================
DSTK DSEG   IHRAM
STACKEND:
            DS     20H          ; Memory stack area = 32 bytes
STACKTOP:                       ; Start address of the memory stack area


;*******************************************************************************
;
;    Servicing interrupts by using unnecessary interrupt sources
;
;*******************************************************************************
XMAIN       CSEG   UNIT
IINIT:
;    If an unnecessary interrupt occurred, the processing branches to this line.
;    The processing then returns to the initial original processing because no
processing is performed here.

     RETI

;*******************************************************************************
;
;    Initialization after RESET
;
;*******************************************************************************
RESET_START:

;-------------------------------------------------------------------------------
;    Disable interrupts
;-------------------------------------------------------------------------------
```

```
     DI                         ; Disable interrupts

;-------------------------------------------------------------------------------
;    Set up the register bank
;-------------------------------------------------------------------------------
     SEL    RB0                 ; Set up the register bank

;-------------------------------------------------------------------------------
;    Specify the ROM and RAM sizes
;-------------------------------------------------------------------------------
;    Note that the values to specify vary depending on the model.
;    Enable the settings for the model to use. (The uPD78F0588 is the default model.)
;-------------------------------------------------------------------------------
     ; Setting when using uPD78F0581 or uPD78F0586
     ;MOV   IMS,   #042H        ; Specify the ROM and RAM sizes

     ; Setting when using uPD78F0582 or uPD78F0587
     ;MOV   IMS,   #004H        ; Specify the ROM and RAM sizes

     ; Setting when using uPD78F0583 or uPD78F0588
     MOV    IMS,   #0C8H        ; Specify the ROM and RAM sizes

;-------------------------------------------------------------------------------
;    Initialize the stack pointer
;-------------------------------------------------------------------------------
     MOVW   SP,    #STACKTOP    ; Initialize the stack pointer

;-------------------------------------------------------------------------------
;    Initialize port 0
;-------------------------------------------------------------------------------
     MOV    P0,    #00000011B   ; Set the P00 and P01 output latches to high level
                                ; Set the P02 output latch to low level
     MOV    PM0,   #11111000B   ; Specify P00 to P02 as output ports
                                ; P00: Used to control LED1
                                ; P01: Used to control LED2
                                ; P02: Unused

;-------------------------------------------------------------------------------
;    Initialize port 1
;-------------------------------------------------------------------------------
     MOV    ADPC1, #00000111B   ; Specify P10 to P12 as digital I/O ports
     MOV    P1,    #00000000B   ; Set the P10 to P17 output latches to low level
     MOV    PM1,   #00000000B   ; Specify P10 to P17 as output ports
                                ; P10 to P17: Unused

;-------------------------------------------------------------------------------
;    Initialize port 2
;-------------------------------------------------------------------------------
```

```
     MOV    ADPC0, #11111111B   ; Specify P20 to P27 as digital I/O ports
     MOV    P2,    #00000000B   ; Set the P20 to P27 output latches to low level
     MOV    PM2,   #00000000B   ; Specify P20 to P27 as output ports
                                ; P20 to P27: Unused

;-------------------------------------------------------------------------------
;    Initialize port 3
;-------------------------------------------------------------------------------
     MOV    P3,    #00000000B   ; Set the P30 to P33 output latches to low level
     MOV    PM3,   #11110000B   ; Specify P30 to P33 as output ports
                                ; P30 to P33: Unused

;-------------------------------------------------------------------------------
;    Initialize port 4
;-------------------------------------------------------------------------------
     MOV    P4,    #00000000B   ; Set the P40 to P42 output latches to low level
     MOV    PM4,   #11111000B   ; Specify P40 to P42 as output ports
                                ; P40 to P42: Unused

;-------------------------------------------------------------------------------
;    Initialize port 6
;-------------------------------------------------------------------------------
     MOV    P6,    #00000000B   ; Set the P60 to P63 output latches to low level
     MOV    PM6,   #11110000B   ; Specify P60 to P63 as output ports
                                ; P60 to P63: Unused

;-------------------------------------------------------------------------------
;    Initialize port 7
;-------------------------------------------------------------------------------
     MOV    P7,    #00000000B   ; Set the P70 to P75 output latches to low level
     MOV    PM7,   #11000000B   ; Specify P70 to P75 as output ports
                                ; P70 to P75: Unused

;-------------------------------------------------------------------------------
;    Initialize port 12
;-------------------------------------------------------------------------------
     MOV    P12,   #00000000B   ; Set the P120 output latch to low level
     MOV    PM12,  #11111110B   ; Specify P120 as an output port
                                ; P120 to P125: Unused

;-------------------------------------------------------------------------------
;    Low-voltage detection
;-------------------------------------------------------------------------------
;    The low-voltage detector is used to check whether VDD is 2.7 V or more.
;-------------------------------------------------------------------------------
     ; Set up the low-voltage detector
     SET1   LVIMK               ; Disable the INTLVI interrupt
     CLR1   LVISEL              ; Specify VDD as the detection voltage
```

```
       MOV   LVIS,  #00001001B   ; Set the low-voltage detection level (VLVI) to 2.84
±0.1 V
       CLR1  LVIMD               ; Specify that an interrupt signal is generated when a
low voltage is detected
       SET1  LVION               ; Enable low-voltage detection

       ; Make the system wait until the low-voltage detector stabilizes (10 us or more)
       MOV   B,     #5           ; Specify the number of counts
  HINI100:
       NOP
       DBNZ  B,     $HINI100     ; Has the wait period ended? No,

       ; Make the system wait until VLVI is less than or equal to VDD
  HINI110:
       NOP
       BT    LVIF,  $HINI110     ; VDD < VLVI? Yes,
       CLR1  LVION               ; Stop the low-voltage detector

  ;-------------------------------------------------------------------------------
  ;    Specify the clock frequency
  ;-------------------------------------------------------------------------------
  ;    Specify the clock frequency so that the device can run on the internal high-speed
oscillation clock.
  ;    The XT1 oscillation mode is specified for the subsystem clock pin.
  ;-------------------------------------------------------------------------------
       MOV   OSCCTL,#00010000B   ; Clock operation mode
  ;                  ||||+||+------- Be sure to clear this bit to 0
  ;                  |||| ++-------- RSWOSC/AMPHXT
  ;                  ||||          [XT1 oscillator oscillation mode selection]
  ;                  ||||           00: Low power consumption oscillation
  ;                  ||||           01: Normal oscillation
  ;                  ||||           1x: Ultra-low power consumption oscillation
  ;                  ||++----------- EXCLKS/OSCSELS
  ;                  ||            [Subsystem clock pin operation setting]
  ;                  ||            (P123/XT1,P124/XT2/EXCLKS)
  ;                  ||             Specify the use of the pin in XT1 oscillation mode by
specifying 001 by also using XTSTART
  ;                  ++------------- EXCLK/OSCSEL
  ;                                [High-speed system clock pin operation setting]
  ;                                (P121/X1,P122/X2/EXCLK)
  ;                                 00: Input port
  ;                                 01: X1 oscillation mode
  ;                                 10: Input port
  ;                                 11: External clock input mode

       MOV   PCC,   #00000000B   ; Select the CPU clock (fCPU)
  ;                  |||+|+++------- CSS/PCC2/PCC1/PCC0
  ;                  ||| |         [CPU clock (fCPU) selection]
```

```
  ;                  ||| |          0000:fXP
  ;                  ||| |          0001:fXP/2
  ;                  ||| |          0010:fXP/2^2
  ;                  ||| |          0011:fXP/2^3
  ;                  ||| |          0100:fXP/2^4
  ;                  ||| |          1000:fSUB/2
  ;                  ||| |          1001:fSUB/2
  ;                  ||| |          1010:fSUB/2
  ;                  ||| |          1011:fSUB/2
  ;                  ||| |          1100:fSUB/2
  ;                  ||| |          (Other than the above: Setting prohibited)
  ;                  ||| +---------- Be sure to clear this bit to 0
  ;                  ||+------------ CLS
  ;                  ||             [CPU clock status]
  ;                  |+------------- XTSTART
  ;                  |              [Subsystem clock pin operation setting]
  ;                  |               Specify the use of the pin by also using EXCLKS and
OSCSELS
  ;                  +-------------- Be sure to clear this bit to 0

       MOV    RCM,   #00000010B   ; Select the operating mode of the internal oscillator
  ;                  |||||||+------- RSTOP
  ;                  |||||||        [Internal high-speed oscillator oscillating/stopped]
  ;                  |||||||         0: Internal high-speed oscillator oscillating
  ;                  |||||||         1: Internal high-speed oscillator stopped
  ;                  ||||||+-------- LSRSTOP
  ;                  ||||||         [Internal low-speed oscillator oscillating/stopped]
  ;                  ||||||          0: Internal low-speed oscillator oscillating
  ;                  ||||||          1: Internal low-speed oscillator stopped
  ;                  |+++++--------- Be sure to clear this bit to 0
  ;                  +-------------- RSTS
  ;                                 [Status of internal high-speed oscillator]

       MOV    MOC,   #10000000B   ;  Select  the  operating  mode  of  the  high-speed  system
clock
  ;                  |+++++++------- Be sure to clear this bit to 0
  ;                  +-------------- MSTOP
  ;                                 [Control of high-speed system clock operation]
  ;                                  0: X1 oscillator operating/external clock from
  ;                                     EXCLK pin is enabled
  ;                                  1: X1 oscillator stopped/external clock from
  ;                                     EXCLK pin is disabled

       MOV    MCM,   #00000000B   ; Select the clock to supply
  ;                  |||||+|+------- XSEL/MCM0
  ;                  ||||| |        [Clock supplied to main system and
  ;                  ||||| |         peripheral hardware]
  ;                  ||||| |         00: Main system clock (fXP)
```

```
;                  ||||| |             = internal high-speed oscillation clock (fIH)
;                  ||||| |            Peripheral hardware clock (fPRS)
;                  ||||| |             = internal high-speed oscillation clock (fIH)
;                  ||||| |        01: Main system clock (fXP)
;                  ||||| |             = internal high-speed oscillation clock (fIH)
;                  ||||| |            Peripheral hardware clock (fPRS)
;                  ||||| |             = internal high-speed oscillation clock (fIH)
;                  ||||| |        10: Main system clock (fXP)
;                  ||||| |             = internal high-speed oscillation clock (fIH)
;                  ||||| |            Peripheral hardware clock (fPRS)
;                  ||||| |             = high-speed system clock (fIH)
;                  ||||| |        11: Main system clock (fXP)
;                  ||||| |             = high-speed system clock (fIH)
;                  ||||| |            Peripheral hardware clock (fPRS)
;                  ||||| |             = high-speed system clock (fIH)
;                  ||||| +-------- MCS
;                  |||||         [Main system clock status]
;                  +++++---------- Be sure to clear this bit to 0

;-----------------------------------------------------------------------------
;    Disable peripheral hardware not to be used
;-----------------------------------------------------------------------------
     ; 16-bit timer/event counter 00
     MOV    TMC00, #00000000B   ; Disable the counter

     ; 8-bit timer/event counters 50 and 51
     MOV    TMC50, #00000000B   ; Disable timer 50
     MOV    TMC51, #00000000B   ; Disable timer 51

     ; 8-bit timer H1
     MOV    TMHMD1,      #00000000B   ; Stop the timer

     ; Clock output controller
     MOV    CKS,   #00000000B   ; Stop the clock frequency divider

     ; A/D converter
     MOV    ADM0,  #00000000B   ; Stop A/D conversion

     ; Operational amplifiers
     MOV    AMP0M, #00000000B   ; Stop operational amplifier 0
     MOV    AMP1M, #00000000B   ; Stop operational amplifier 1

     ; Serial interface UART6
     MOV    ASIM6, #00000001B   ; Disable the interface

     ; Serial interface IICA
     MOV    IICACTL0,#00000000B ; Disable the interface
```

```
       ; Serial interfaces CSI10 and CSI11
       MOV    CSIM10,      #00000000B   ; Disable CSI10
       MOV    CSIM11,      #00000000B   ; Disable CSI11

       ; Interrupts (The interrupts to be used are enabled later)
       MOVW   MK0,   #0FFFFH      ; Disable all interrupts
       MOVW   MK1,   #0FFFFH      ;
       MOV    EGPCTL0,#00000000B  ; Disable the detection of all external interrupts
       MOV    EGPCTL1,#00000000B  ;

       ; Key interrupts
       MOV    KRM,   #00000000B   ; Disable all key interrupts

  ;-------------------------------------------------------------------------------
  ;    Set up the real-time counter
  ;-------------------------------------------------------------------------------
  ;    - Making the system wait until the subsystem clock stabilizes by using 8-bit timer
H0 (about 1 second)
  ;    - Specifying 8:59:50 a.m. as the count start time and 9:00:00 a.m. as the alarm
time
  ;    - Setting fixed-interval interrupts (generated at 0.5 second intervals) and alarm
interrupts
  ;-------------------------------------------------------------------------------
       ; Make the system wait until the subsystem clock stabilizes by using 8-bit timer H0
(about 1 second)
       MOV    TMHMD0,      #01000000B   ; Count clock: fPRS/2^10
       MOV    CMP00, #(245-1)     ; Interval time: 31.36 ms (= 245 / fPRS/2^10)
       MOV    B,     #32          ; Oscillation stabilization time: About 1 second (=
31.36 ms * 32)
       CLR1   TMIFH0              ; Clear the INTTMH0 interrupt request
       SET1   TMMKH0              ; Disable the INTTMH0 interrupt
       SET1   TMHE0               ; Start 8-bit timer H0
  HINI200:
       NOP
       BF     TMIFH0,      $HINI200    ; Has an INTTMH0 interrupt request been issued? No,
       CLR1   TMIFH0              ; Clear the INTTMH0 interrupt request
       DBNZ   B,     $HINI200     ; Has the oscillation stabilization time elapsed? No,
       CLR1   TMHE0               ; Stop 8-bit timer H0

       ; Control of real-time counter control clock
       MOV    PER0,  #10000000B
  ;                  |+++++++------- Be sure to clear this bit to 0
  ;                  +-------------- RTCEN
  ;                                 [Real-time counter control clock]
  ;                                  0: Stop supply of control clock
  ;                                  1: Supply control clock

       ; Real-time counter operation setting
```

```
       CLR1   RTCE                ; Stop real-time counter operation
       MOV    RTCC0, #00000001B
  ;                  |||||+++------- CT2/CT1/CT0
  ;                  |||||         [Fixed-interval interrupt (INTRTC) selection]
  ;                  |||||          000: Does not use fixed-interval interrupt function
  ;                  |||||          001:  Once  per  0.5  s  (synchronized  with  second  count
up)
  ;                  |||||          010: Once per 1 s (same time as second count up)
  ;                  |||||          011: Once per 1 m (second 00 of every minute)
  ;                  |||||          100:  Once  per  1  hour  (minute  00  and  second  00  of
every hour)
  ;                  |||||          101:  Once  per  1  day  (hour  00,  minute  00,  and  second
00 of every day)
  ;                  |||||          11x: Once per 1 month (Day 1, hour 00 a.m., minute 00,
and second 00 of every month)
  ;                  ||||+---------- AMPM
  ;                  ||||          [Selection of 12-/24-hour system]
  ;                  ||||           0: 12-hour system (a.m. and p.m. are displayed)
  ;                  ||||           1: 24-hour system
  ;                  |||+----------- RCLOE0
  ;                  |||           [RTCCL pin output control]
  ;                  |||            0: Disable output of RTCCL pin (32.768 kHz)
  ;                  |||            1: Enable output of RTCCL pin (32.768 kHz)
  ;                  ||+------------ RCLOE1
  ;                  ||            [RTC1HZ pin output control]
  ;                  ||             0: Disable output of RTC1HZ pin (1 Hz)
  ;                  ||             1: Enable output of RTC1HZ pin (1 Hz)
  ;                  |+------------- Be sure to clear this bit to 0
  ;                  +-------------- RTCE
  ;                                [Real-time counter operation control]
  ;                                 0: Stop counter operation
  ;                                 1: Start counter operation

       ; Specify the count start time (January 1st, 2000, Saturday, 8:59:50 a.m.)
       MOV    SEC,   #50H         ; seconds: 50
       MOV    MIN,   #59H         ; minutes: 59
       MOV    HOUR,  #08H         ; hours: 08
       MOV    WEEK,  #06H         ; day of the week: Saturday
       MOV    DAY,   #01H         ; day: 01
       MOV    MONTH, #01H         ; month: 01
       MOV    YEAR,  #00H         ; year: 00

       ; Set up the generation of alarm interrupts
       CLR1   WALE                ; Disable matching
       SET1   WALIE               ; Generate an interrupt upon an alarm time match

       ; Specify the alarm time (9:00 a.m. every day)
       MOV    ALARMWM,#00H        ; minutes: 00
```

```
      MOV    ALARMWH,#09H        ; hours: 09
      MOV    ALARMWW,#01111111B  ; day of the week: all days

      SET1   WALE                ; Enable alarm time matching

      CLR1   RTCIF               ; Clear the INTRTC interrupt request
      CLR1   RTCMK               ; Enable the INTRTC interrupt

      SET1   RTCE                ; Start the real-time counter

      ; Make the system wait before entering the STOP mode immediately after starting the
real-time counter
      MOV    B,     #62          ; * When entering the STOP mode immediately
 HINI210:                        ;   after setting RTCE to 1, first wait for
      NOP                        ;   two cycles of the subsystem clock (about 62 us)
      DBNZ   B,     $HINI210     ;   or more.

 ;-------------------------------------------------------------------------------
 ;    Enable interrupts
 ;-------------------------------------------------------------------------------
      EI                         ; Enable interrupts

      BR     MMAIN_LOOP          ; Go to the main loop

 ;*******************************************************************************
 ;
 ;    Main loop
 ;
 ;*******************************************************************************
 MMAIN_LOOP:
      ; Make the system wait for an interrupt
      STOP                       ; Enter the STOP mode (Exit the STOP mode by generating
an INTRTC interrupt)
      BR     MMAIN_LOOP          ; Go to the start of the main loop

 ;*******************************************************************************
 ;
 ;    INTRTC interrupt servicing
 ;    (by using the fixed-interval signal of the real-time counter or alarm time match
detection)
 ;
 ;*******************************************************************************
 IINTRTC:
      SEL    RB1                 ; Switch the register bank
      XOR    P0,    #00000001B   ; Toggle LED1
      BF     WAFG,  $HRTC800     ; Has an alarm time match been detected? No,

      ; If an alarm time match is detected
```

```
     CLR1   WAFG                ; Clear the alarm detection status flag
     CLR1   P0.1                ; Turn on LED2

     ; * To add other processing when an alarm time match has been detected,
     ;   specify it here.

HRTC800:
     RETI
end
```

- main.c (C language version)

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

  NEC Electronics     78K0/KC2-L Series

*******************************************************************************
  78K0/KC2-L Series   Sample Program (Real-Time Counter)
*******************************************************************************
  Fixed-Interval Interrupts and Alarm Interrupts
*******************************************************************************
<<History>>
  2009.1.-- Release
*******************************************************************************

<<Overview>>

This sample program presents an example of using the real-time counter.  In the
sample program, LEDs (LED1 and LED2) are controlled by using the real-time counter
to generate fixed-interval interrupts and alarm interrupts.


 <Primary initial settings>

 (Option byte settings)
 - Allowing the internal low-speed oscillator to be programmed to stop
 - Disabling the watchdog timer
 - Setting the internal high-speed oscillation clock frequency to 8 MHz
 - Disabling LVI from being started by default
 (Settings during initialization immediately after a reset ends)
 - Specifying the ROM and RAM sizes
 - Setting up I/O ports
   → Specifying P00 and P01 to be used to control the LEDs (LED1 and LED2)
 - Checking whether VDD is 2.7 V or more by using the low-voltage detector
 - Specifying that the CPU clock runs on the internal high-speed oscillation clock (8
MHz)
 - Specifying the XT1 oscillation mode for the subsystem clock pin
 - Stopping the internal low-speed oscillator
 - Disabling peripheral hardware not to be used
 - Setting up the real-time counter
   → Making the system wait until the subsystem clock stabilizes by using 8-bit timer H0
(about 1 second)
   → Specifying 8:59:50 a.m. as the count start time and 9:00:00 a.m. as the alarm time
   → Setting fixed-interval interrupts (generated at 0.5 second intervals) and alarm
interrupts
 - Enabling interrupts


 <LED control timing>
```

```
   +-------------------------------------------+
   | LED         | LED control timing          |
   |-------------------------------------------|
   | LED1 (P00)  | Blink every second.         |
   | LED2 (P01)  | Turn on at the alarm time.  |
   +-------------------------------------------+
   * The LEDs turn off if 1 is output from the ports and turn on if 0 is output from the
ports.


   <I/O port settings>
    Output: P00, P01
    * Set all unused ports that can be specified as output ports as output ports.

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


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

   Preprocessing directive (#pragma)

  ============================================================================*/
  #pragma SFR             /* SFR names can be described at the C source level   */
  #pragma DI              /* DI instructions 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 STOP            /* STOP instructions can be described at the C source level */
  #pragma interrupt INTRTC fn_intrtc RB1/* Declare the interrupt function: INTRTC */


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

   Initialization after RESET

  *****************************************************************************/
  void hdwinit( void )
  {
   unsigned char ucCounter;   /* Count variable */

  /*----------------------------------------------------------------------------
   Disable interrupts
  ----------------------------------------------------------------------------*/
   DI();               /* Disable interrupts */

  /*----------------------------------------------------------------------------
   Specify the ROM and RAM sizes
  ----------------------------------------------------------------------------
```

```
  Note that the values to specify vary depending on the model.
  Enable the settings for the model to use. (The uPD78F0588 is the default model.)
--------------------------------------------------------------------------------*/
  /* Setting when using uPD78F0581 or uPD78F0586 */
  /*IMS =   0x42;*/          /* Specify the ROM and RAM sizes */

  /* Setting when using uPD78F0582 or uPD78F0587 */
  /*IMS =   0x04;*/          /* Specify the ROM and RAM sizes */

  /* Setting when using uPD78F0583 or uPD78F0588 */
  IMS =  0xC8;         /* Specify the ROM and RAM sizes */

/*--------------------------------------------------------------------------------
  Initialize port 0
--------------------------------------------------------------------------------*/
  P0     = 0b00000011; /* Set the P00 and P01 output latches to high level */
                       /* Set the P02 output latch to low level */
  PM0    = 0b11111000; /* Specify P00 to P02 as output ports */
                       /* P00: Used to control LED1 */
                       /* P01: Used to control LED2 */
                       /* P02: Unused */

/*--------------------------------------------------------------------------------
  Initialize port 1
--------------------------------------------------------------------------------*/
  ADPC1  = 0b00000111; /* Specify P10 to P12 as digital I/O ports */
  P1     = 0b00000000; /* Set the P10 to P17 output latches to low level */
  PM1    = 0b00000000; /* Specify P10 to P17 as output ports */
                       /* P10 to P17: Unused */

/*--------------------------------------------------------------------------------
  Initialize port 2
--------------------------------------------------------------------------------*/
  ADPC0  = 0b11111111; /* Specify P20 to P27 as digital I/O ports */
  P2     = 0b00000000; /* Set the P20 to P27 output latches to low level */
  PM2    = 0b00000000; /* Specify P20 to P27 as output ports */
                       /* P20 to P27: Unused */

/*--------------------------------------------------------------------------------
  Initialize port 3
--------------------------------------------------------------------------------*/
  P3     = 0b00000000; /* Set the P30 to P33 output latches to low level */
  PM3    = 0b11110000; /* Specify P30 to P33 as output ports */
                       /* P30 to P33: Unused */

/*--------------------------------------------------------------------------------
  Initialize port 4
--------------------------------------------------------------------------------*/
```

```
    P4     = 0b00000000; /* Set the P40 to P42 output latches to low level */
    PM4    = 0b11111000; /* Specify P40 to P42 as output ports */
                         /* P40 to P42: Unused */

  /*--------------------------------------------------------------------------------
    Initialize port 6
  --------------------------------------------------------------------------------*/
    P6     = 0b00000000; /* Set the P60 to P63 output latches to low level */
    PM6    = 0b11110000; /* Specify P60 to P63 as output ports */
                         /* P60 to P63: Unused */

  /*--------------------------------------------------------------------------------
    Initialize port 7
  --------------------------------------------------------------------------------*/
    P7     = 0b00000000; /* Set the P70 to P75 output latches to low level */
    PM7    = 0b11000000; /* Specify P70 to P75 as output ports */
                         /* P70 to P75: Unused */

  /*--------------------------------------------------------------------------------
    Initialize port 12
  --------------------------------------------------------------------------------*/
    P12    = 0b00000000; /* Set the P120 output latch to low level */
    PM12   = 0b11111110; /* Specify P120 as an output port */
                         /* P120 to P125: Unused */

  /*--------------------------------------------------------------------------------
    Low-voltage detection
  ----------------------------------------------------------------------------------
    The low-voltage detector is used to check whether VDD is 2.7 V or more.
  --------------------------------------------------------------------------------*/
    /* Set up the low-voltage detector */
    LVIMK  = 1;          /* Disable the INTLVI interrupt */
    LVISEL = 0;          /* Specify VDD as the detection voltage */
    LVIS   = 0b00001001; /* Set the low-voltage detection level (VLVI) to 2.84 ±0.1 V */
    LVIMD  = 0;          /* Specify that an interrupt signal is generated when a low
voltage is detected */
    LVION  = 1;          /* Enable low-voltage detection */

    /* Make the system wait until the low-voltage detector stabilizes (10 us or more) */
    for( ucCounter = 0; ucCounter < 2; ucCounter++ ){
       NOP();
    }

    /* Make the system wait until VLVI is less than or equal to VDD */
    while(LVIF){
       NOP();
    }
    LVION  = 0;          /* Stop the low-voltage detector */
```

```
  /*--------------------------------------------------------------------------------
   Specify the clock frequency
  --------------------------------------------------------------------------------
   Specify the clock frequency so that the device can run on the internal high-speed
oscillation clock.
   The XT1 oscillation mode is specified for the subsystem clock pin.
  --------------------------------------------------------------------------------*/
   OSCCTL = 0b00010000; /* Clock operation mode */
   /*         ||||+||+---- Be sure to clear this bit to 0 */
   /*         |||| ++----- RSWOSC/AMPHXT */
   /*         ||||        [XT1 oscillator oscillation mode selection] */
   /*         ||||         00: Low power consumption oscillation */
   /*         ||||         01: Normal oscillation */
   /*         ||||         1x: Ultra-low power consumption oscillation */
   /*         ||++-------- EXCLKS/OSCSELS */
   /*         ||          [Subsystem clock pin operation setting] */
   /*         ||          (P123/XT1,P124/XT2/EXCLKS) */
   /*         ||           Specify the use of the pin in XT1 oscillation mode by
specifying 001 by also using XTSTART */
   /*         ++---------- EXCLK/OSCSEL */
   /*                     [High-speed system clock pin operation setting] */
   /*                      (P121/X1,P122/X2/EXCLK) */
   /*                      00: Input port */
   /*                      01: X1 oscillation mode */
   /*                      10: Input port */
   /*                      11: External clock input mode */

   PCC    = 0b00000000; /* Select the CPU clock (fCPU) */
   /*         |||+|+++---- CSS/PCC2/PCC1/PCC0 */
   /*         ||| |       [CPU clock (fCPU) selection] */
   /*         ||| |        0000:fXP */
   /*         ||| |        0001:fXP/2 */
   /*         ||| |        0010:fXP/2^2 */
   /*         ||| |        0011:fXP/2^3 */
   /*         ||| |        0100:fXP/2^4 */
   /*         ||| |        1000:fSUB/2 */
   /*         ||| |        1001:fSUB/2 */
   /*         ||| |        1010:fSUB/2 */
   /*         ||| |        1011:fSUB/2 */
   /*         ||| |        1100:fSUB/2 */
   /*         ||| |        (Other than the above: Setting prohibited) */
   /*         ||| +------- Be sure to clear this bit to 0 */
   /*         ||+--------- CLS */
   /*         ||          [CPU clock status] */
   /*         |+---------- XTSTART */
   /*         |           [Subsystem clock pin operation setting] */
   /*         |            Specify the use of the pin by also using EXCLKS and OSCSELS */
```

```
   /*          +----------- Be sure to clear this bit to 0 */

   RCM    = 0b00000010; /* Select the operating mode of the internal oscillator */
   /*          |||||||+---- RSTOP */
   /*          |||||||     [Internal high-speed oscillator oscillating/stopped] */
   /*          |||||||      0: Internal high-speed oscillator oscillating */
   /*          |||||||      1: Internal high-speed oscillator stopped */
   /*          ||||||+----- LSRSTOP */
   /*          ||||||      [Internal low-speed oscillator oscillating/stopped] */
   /*          ||||||       0: Internal low-speed oscillator oscillating */
   /*          ||||||       1: Internal low-speed oscillator stopped */
   /*          |+++++------ Be sure to clear this bit to 0 */
   /*          +----------- RSTS */
   /*                      [Status of internal high-speed oscillator] */

   MOC    = 0b10000000; /* Select the operating mode of the high-speed system clock */
   /*          |+++++++---- Be sure to clear this bit to 0 */
   /*          +----------- MSTOP */
   /*                      [Control of high-speed system clock operation] */
   /*                       0: X1 oscillator operating/external clock from EXCLK pin is
enabled */
   /*                       1: X1 oscillator stopped/external clock from EXCLK pin is
disabled */

   MCM    = 0b00000000; /* Select the clock to supply */
   /*          |||||+|+---- XSEL/MCM0 */
   /*          ||||| |        [Clock supplied to main system and peripheral hardware] */
   /*          ||||| |         00: Main system clock (fXP) */
   /*          ||||| |              = internal high-speed oscillation clock (fIH) */
   /*          ||||| |             Peripheral hardware clock (fPRS) */
   /*          ||||| |              = internal high-speed oscillation clock (fIH) */
   /*          ||||| |         01: Main system clock (fXP) */
   /*          ||||| |              = internal high-speed oscillation clock (fIH) */
   /*          ||||| |             Peripheral hardware clock (fPRS) */
   /*          ||||| |              = internal high-speed oscillation clock (fIH) */
   /*          ||||| |         10: Main system clock (fXP) */
   /*          ||||| |              = internal high-speed oscillation clock (fIH) */
   /*          ||||| |             Peripheral hardware clock (fPRS) */
   /*          ||||| |              = high-speed system clock (fIH) */
   /*          ||||| |         11: Main system clock (fXP) */
   /*          ||||| |              = high-speed system clock (fIH) */
   /*          ||||| |             Peripheral hardware clock (fPRS) */
   /*          ||||| |              = high-speed system clock (fIH) */
   /*          ||||| +----- MCS */
   /*          |||||       [Main system clock status] */
   /*          +++++------- Be sure to clear this bit to 0 */

  /*------------------------------------------------------------------------------
```

```
    Disable peripheral hardware not to be used
  ---------------------------------------------------------------------------*/
    /* 16-bit timer/event counter 00 */
    TMC00  = 0b00000000; /* Disable the counter */

    /* 8-bit timer/event counters 50 and 51 */
    TMC50  = 0b00000000; /* Disable timer 50 */
    TMC51  = 0b00000000; /* Disable timer 51 */

    /* 8-bit timer H1 */
    TMHMD1 = 0b00000000; /* Stop timer H1 */

    /* Clock output controller */
    CKS    = 0b00000000; /* Stop the clock frequency divider */

    /* A/D converter */
    ADM0   = 0b00000000; /* Stop A/D conversion */

    /* Operational amplifiers */
    AMP0M  = 0b00000000; /* Stop operational amplifier 0 */
    AMP1M  = 0b00000000; /* Stop operational amplifier 1 */

    /* Serial interface UART6 */
    ASIM6  = 0b00000001; /* Disable the interface */

    /* Serial interface IICA */
    IICACTL0 = 0b00000000;  /* Disable the interface */

    /* Serial interfaces CSI10 and CSI11 */
    CSIM10 = 0b00000000; /* Disable CSI10 */
    CSIM11 = 0b00000000; /* Disable CSI11 */

    /* Interrupts (The interrupts to be used are enabled later) */
    MK0    = 0xFFFF;     /* Disable all interrupts */
    MK1    = 0xFFFF;
    EGPCTL0    = 0b00000000; /* Disable the detection of all external interrupts */
    EGPCTL1    = 0b00000000;

    /* Key interrupts */
    KRM    = 0b00000000; /* Disable all key interrupts */

  /*---------------------------------------------------------------------------
    Set up the real-time counter
  ---------------------------------------------------------------------------
   - Making the system wait until the subsystem clock stabilizes by using 8-bit timer H0
(about 1 second)
   - Specifying 8:59:50 a.m. as the count start time and 9:00:00 a.m. as the alarm time
   -  Setting  fixed-interval  interrupts  (generated  at  0.5  second  intervals)  and  alarm
```

```
interrupts
  ------------------------------------------------------------------------------*/
    /* Make the system wait until the subsystem clock stabilizes by using 8-bit timer H0
(about 1 second) */
    TMHMD0 = 0b01000000;/* Count clock: fPRS/2^10 */
    CMP00  = (245-1);   /* Interval time: 31.36 ms (= 245 / fPRS/2^10) */
    TMIFH0 = 0;         /* Clear the INTTMH0 interrupt request */
    TMMKH0 = 1;         /* Disable the INTTMH0 interrupt */
    TMHE0  = 1;         /* Start 8-bit timer H0 */

    /* Oscillation stabilization time: About 1 second (= 31.36 ms * 32) */
    for( ucCounter = 0; ucCounter < 32; ucCounter++ ){
        /* Make the system wait for an interrupt */
        while( !TMIFH0 ){
            NOP();
        }
        TMIFH0 = 0;  /* Clear the INTTMH0 interrupt request */
    }
    TMHE0  = 0;         /* Stop 8-bit timer H0 */

    /* Control of real-time counter control clock */
    PER0   = 0b10000000;
    /*         |+++++++---- Be sure to clear this bit to 0 */
    /*         +----------- RTCEN: */
    /*                     [Real-time counter control clock] */
    /*                      0: Stop supply of control clock */
    /*                      1: Supply control clock */

    /* Real-time counter operation setting */
    RTCE   = 0;         /* Stop real-time counter operation */
    RTCC0  = 0b00000001;
    /*         |||||+++---  CT2/CT1/CT0                                               */
    /*         |||||      [Fixed-interval interrupt (INTRTC) selection]               */
    /*         |||||       000: Does not use fixed-interval interrupt function        */
    /*         |||||       001: Once per 0.5 s (synchronized with second count up)    */
    /*         |||||       010: Once per 1 s (same time as second count up)           */
    /*         |||||       011: Once per 1 m (second 00 of every minute)              */
    /*         |||||       100: Once per 1 hour (minute 00 and second 00 of every hour) */
    /*         |||||       101: Once per 1 day (hour 00, minute 00, and second 00 of every
day)   */
    /*         |||||       11x: Once per 1 month (Day 1, hour 00 a.m., minute 00, and
second 00 of every month)  */
    /*         ||||+------  AMPM                                                      */
    /*         ||||       [Selection of 12-/24-hour system]                           */
    /*         ||||        0: 12-hour system (a.m. and p.m. are displayed)            */
    /*         ||||        1: 24-hour system                                          */
    /*         |||+-------  RCLOE0                                                    */
    /*         |||        [RTCCL pin output control]                                  */
```

```
    /*         |||        0: Disable output of RTCCL pin (32.768 kHz)               */
    /*         |||        1: Enable output of RTCCL pin (32.768 kHz)                */
    /*         ||+--------  RCLOE1                                                  */
    /*         ||        [RTC1HZ pin output control]                                */
    /*         ||         0: Disable output of RTC1HZ pin (1 Hz)                    */
    /*         ||         1: Enable output of RTC1HZ pin (1 Hz)                     */
    /*         |+---------  Be sure to clear this bit to 0                          */
    /*         +----------  RTCE                                                    */
    /*                   [Real-time counter operation control]                      */
    /*                    0: Stop counter operation                                 */
    /*                    1: Start counter operation                                */

    /* Specify the count start time (January 1st, 2000, Saturday, 8:59:50 a.m.) */
    SEC    = 0x50;          /* seconds: 50 */
    MIN    = 0x59;          /* minutes: 59 */
    HOUR   = 0x08;          /* hours: 08 */
    WEEK   = 0x06;          /* day of the week: Saturday */
    DAY    = 0x01;          /* day: 01 */
    MONTH  = 0x01;          /* month: 01 */
    YEAR   = 0x00;          /* year: 00 */

    /* Set up the generation of alarm interrupts */
    WALE   = 0;          /* Disable matching */
    WALIE  = 1;          /* Generate an interrupt upon an alarm time match */

    /* Specify the alarm time (9:00 a.m. every day) */
    ALARMWM   = 0x00;          /* minutes: 00 */
    ALARMWH   = 0x09;          /* hours: 09 */
    ALARMWW   = 0b01111111;    /* day of the week: all days */

    WALE   = 1;          /* Enable alarm time matching */

    RTCIF  = 0;          /* Clear the INTRTC interrupt request */
    RTCMK  = 0;          /* Enable the INTRTC interrupt */

    RTCE   = 1;          /* Start the real-time counter */

    /* Make the system wait before entering the STOP mode immediately after starting the
real-time counter */
    /* * When entering the STOP mode immediately after setting  */
    /*   RTCE to 1, first wait for two cycles of the subsystem  */
    /*   clock (about 62 us) or more.                           */
    for( ucCounter = 0; ucCounter < 22; ucCounter++ ){
       NOP();
    }

  /*-------------------------------------------------------------------------------
    Enable interrupts
```

```
  -----------------------------------------------------------------------------*/
    EI();               /* Enable interrupts */

  }


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

    Main loop

  *****************************************************************************/
  void main(void)
  {
    while (1){
        STOP();   /* Enter the STOP mode (Exit the STOP mode by generating an INTRTC
interrupt) */
    }
  }

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

    INTRTC interrupt servicing
    (by using the fixed-interval signal of the real-time counter or alarm time match
detection)

  *****************************************************************************/
  __interrupt void fn_intrtc(void)
  {
    P0 ^= 0b00000001;   /* Toggle LED1 */

    /* If an alarm time match is detected */
    if( WAFG ){
        WAFG = 0; /* Clear the alarm detection status flag */
        P0.1 = 0; /* Turn on LED2 */

        /* * To add other processing when an alarm time match has been detected, */
        /*   specify it here.                                                    */
    }
  }
```

APPENDIX B USING 78K0/KC2-L 44-PIN PRODUCTS

All 78K0/KC2-L sample programs are intended for 48-pin products. To use a 78K0/KC2-L sample program for a 44-pin product, specify the following settings:

(1) Initial settings of ports

- Setting up port 0
  Change the value of bit 2 of port mode register 0 (PM0) from “0” to “1”.

- Setting up port 4
  Change the value of bit 2 of port mode register 4 (PM4) from “0” to “1”.

- Setting up port 7
  Change the values of bits 5 and 4 of port mode register 7 (PM7) from “00” to “11”.

(2) Disabling unused peripheral hardware

Delete the instruction used to set up the clock output selection register (CKS).

APPENDIX C REVISION HISTORY

Edition	Date Published	Page	Revision
1st edition	September 2009	–	–

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