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.

RENESAS

Application Note

78K0/Kx2-L

Sample Program (Serial Interface IICA)

Master Communication

This document describes an operation overview of the sample program and how to use it, as well as how to set up and use serial interface IICA. In the sample program, 16 bytes of data are transmitted and received via the I^2C bus in master operation. Communication starts at intervals of about 5 ms, alternating between transmission and reception.

Target devices

78K0/KY2-L microcontroller
78K0/KA2-L microcontroller
78K0/KB2-L microcontroller
78K0/KC2-L microcontroller

CONTENTS

Document No. U19690EJ1V0AN00 (1st edition)
Date Published August 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.

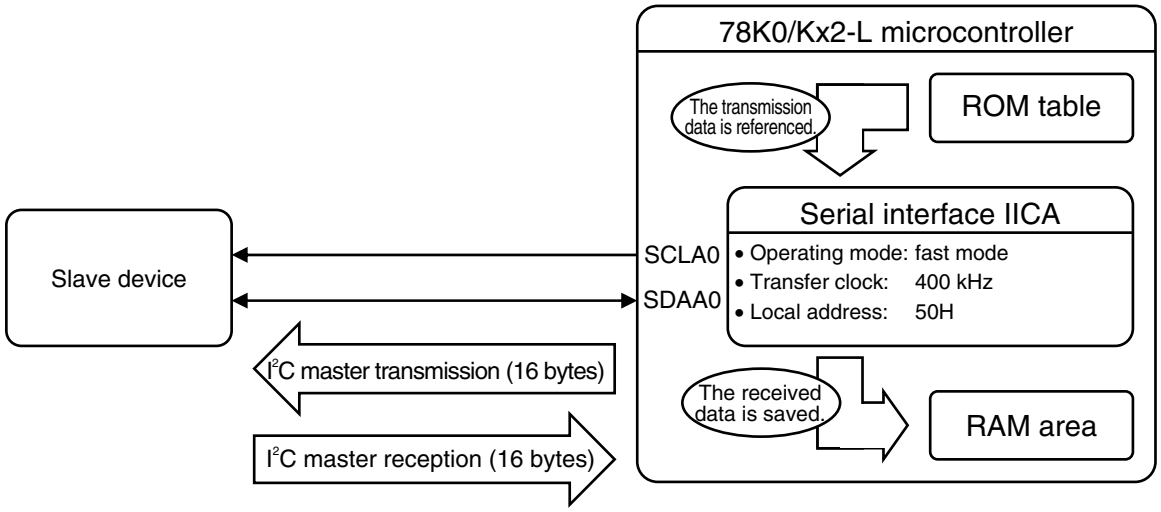
(2) "NEC Electronics products" means any product developed or manufactured by or for NEC Electronics (as defined above).

M8E0904E

CHAPTER 1 OVERVIEW

This sample program shows an example of using serial interface IICA. 16 bytes of data are transmitted and received via the I²C bus in master operation. Communication starts at intervals of about 5 ms, alternating between transmission and reception.

[Operation overview]



Caution For a definition of the I²C bus, refer to the 78K0/Kx2-L User's Manual.

1.1 Primary Initial Settings

The primary initial settings are as follows.

<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
- Checking whether V_{DD} is 2.7 V or more by using the low-voltage detector[Note 1]
- Specifying that the CPU clock and peripheral hardware clock run on the internal high-speed oscillation clock (8 MHz)
- Disabling peripheral hardware not to be used
- Setting up serial interface IICA
  - Specifying fast mode as the operating mode and setting the transfer clock frequency to 400 kHz
  - Specifying 50H as the local address
  - Specifying that P60/SCLA0 and P61/SDAA0 are used for the I^2C bus
  - Enabling the INTIICA0 interrupt[Note 2]
- Specifying that 8-bit timer H1[Note 3] runs in intervals of about 5 ms to trigger I^2C communication

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

2. In this sample program, the HALT mode is entered while the system waits for data communication to end, and the HALT mode is exited when the INTIICA0 interrupt is generated at the end of data communication. When adding other interrupts to this sample program, make sure that these interrupts do not affect the HALT mode from being exited when the INTIICA0 interrupt occurs.

3. For details about 8-bit timer H1, refer to the 78K0/Kx2-L User's Manual.

1.2 Processing After Main Loop

After the initial settings have been specified, the STOP mode is entered. Next, the STOP mode is exited when the INTTMH1 interrupt is generated at intervals of about 5 ms, and I^2C transmission starts. The transmission data is assumed to be 16 bytes and the slave address to which the data is transmitted is assumed to be A0H. If transmission does not end normally, transmission restarts up to three times. After transmission ends, the STOP mode is entered again. The STOP mode is exited when the INTTMH1 interrupt is generated at intervals of about 5 ms, and I^2C reception starts. 16 bytes of data are received and then saved in RAM.

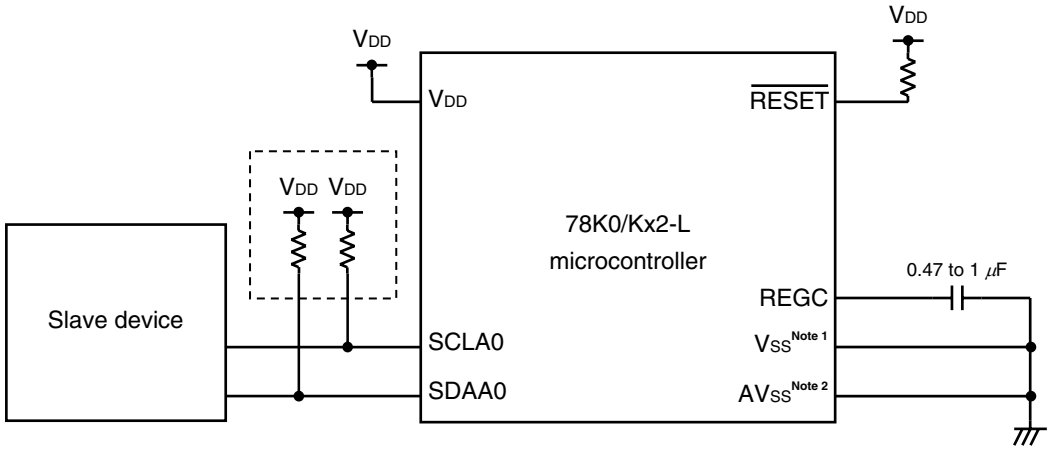
In the same way, transmission and reception alternate at intervals of about 5 ms.

CHAPTER 2 CIRCUIT DIAGRAM

This chapter provides a circuit diagram used in this sample program.

2.1 Circuit Diagram

A circuit diagram is shown below.



Notes 1. This is shared with AVSS in the 78K0/KY2-L and 78K0/KA2-L.

2. This is provided only in the 78K0/KB2-L and 78K0/KC2-L.

Cautions 1. Use the microcontroller at a voltage in the range of 2.94 V ≤ VDD ≤ 5.5 V.

2. Connect REGC to VSS via a capacitor (0.47 to 1 μF).

3. For the 78K0/KY2-L and 78K0/KA2-L, VSS is also used as the ground potential for the A/D converter. Be sure to connect VSS to a stable GND.

4. Make the AVSS pin have the same potential as VSS and connect it directly to GND (only for the 78K0/KB2-L and 78K0/KC2-L microcontrollers).

5. Connect the AVREF pin directly to VDD.

6. 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 VDD or VSS via a resistor.**

7. Adjust the resistance of the pull-up resistors connected to the serial clock line and serial data bus line (enclosed in the dotted lines above) in accordance with the voltage and capacitance of the I^2C bus and the transfer clock. In this sample program, resistors with a resistance of 2 to 10 kΩ are used.

8. In this sample program, the P121/X1/TOOLC0 and P122/X2/EXCLK/TOOLD0 pins are used for on-chip debugging.

2.2 Used Device Other than Microcontroller

The following device is used in addition to the microcontroller:

(1) Slave device

A device that performs slave transmission and reception is used as the other party of I^2C master communication.

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	
			
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_IICAM.prw	Work space file for integrated development environment PM+		●
Kx2-L_IICAM.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  : Only the source file is included.

 : 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.

(1) Peripheral hardware

- Serial interface IICA: Performs I^2C master communication.
- 8-bit timer H1: Triggers I^2C communication.
- Low-voltage detector: Checks whether V_{DD} is 2.7 V or more.

(2) Pin functions

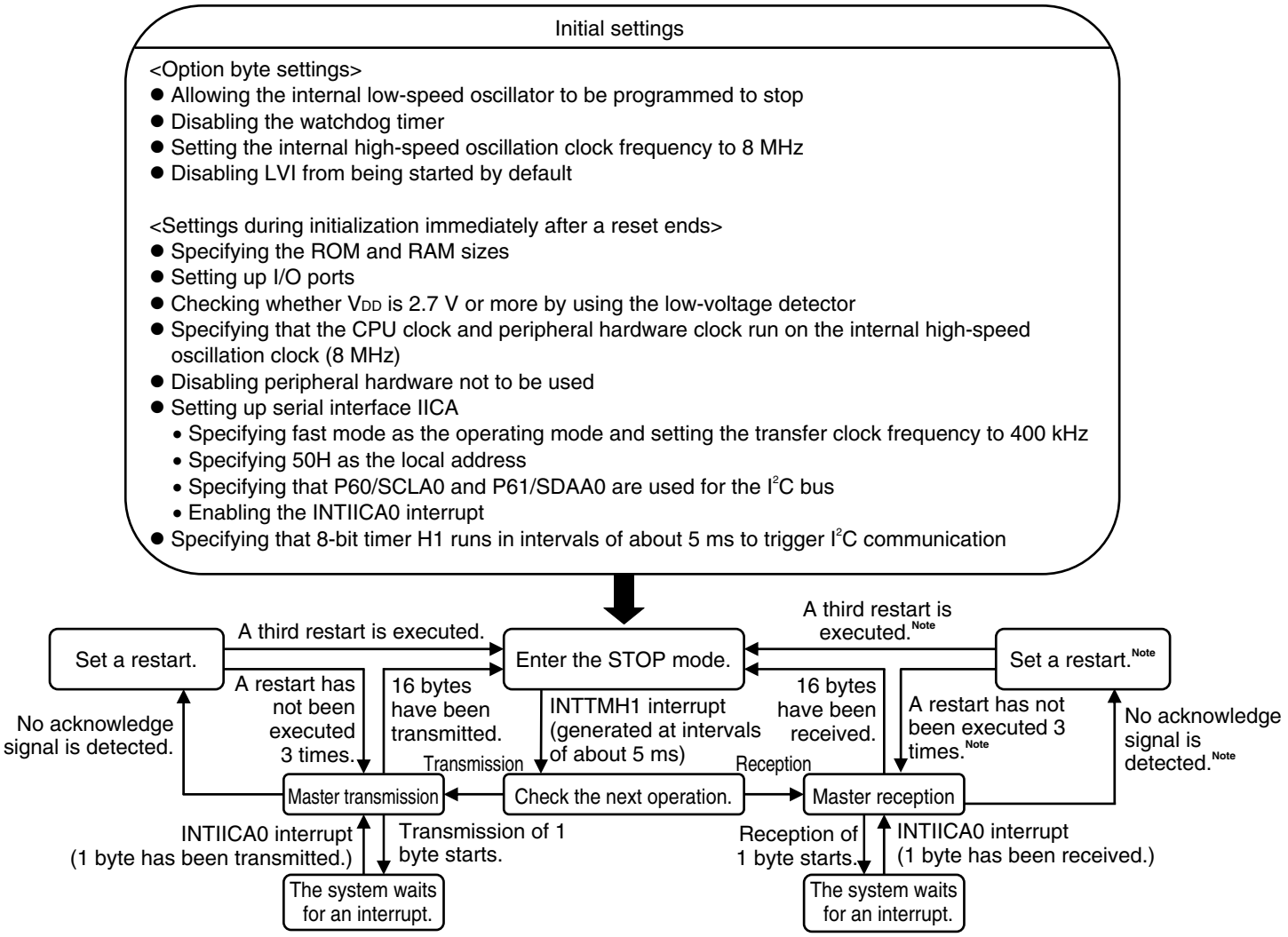
- SCLA0/P60: Used as the I^2C serial clock pin.
- SDAA0/P61: Used as the I^2C serial data bus pin.

3.3 Initial Settings and Operation Overview

In this sample program, initial settings including the selection of the clock frequency, setting of the I/O ports, and setting of serial interface IICA are performed. After the initial settings have been specified, the STOP mode is entered. Next, the STOP mode is exited when the INTTMH1 interrupt is generated at intervals of about 5 ms, and I^2C transmission starts. The transmission data is assumed to be 16 bytes and the slave address to which the data is transmitted is assumed to be A0H. If transmission does not end normally, transmission restarts up to three times. After transmission ends, the STOP mode is entered again. The STOP mode is exited when the INTTMH1 interrupt is generated at intervals of about 5 ms, and I^2C reception starts. 16 bytes of data are received and then saved in RAM.

In the same way, transmission and reception alternate at intervals of about 5 ms.

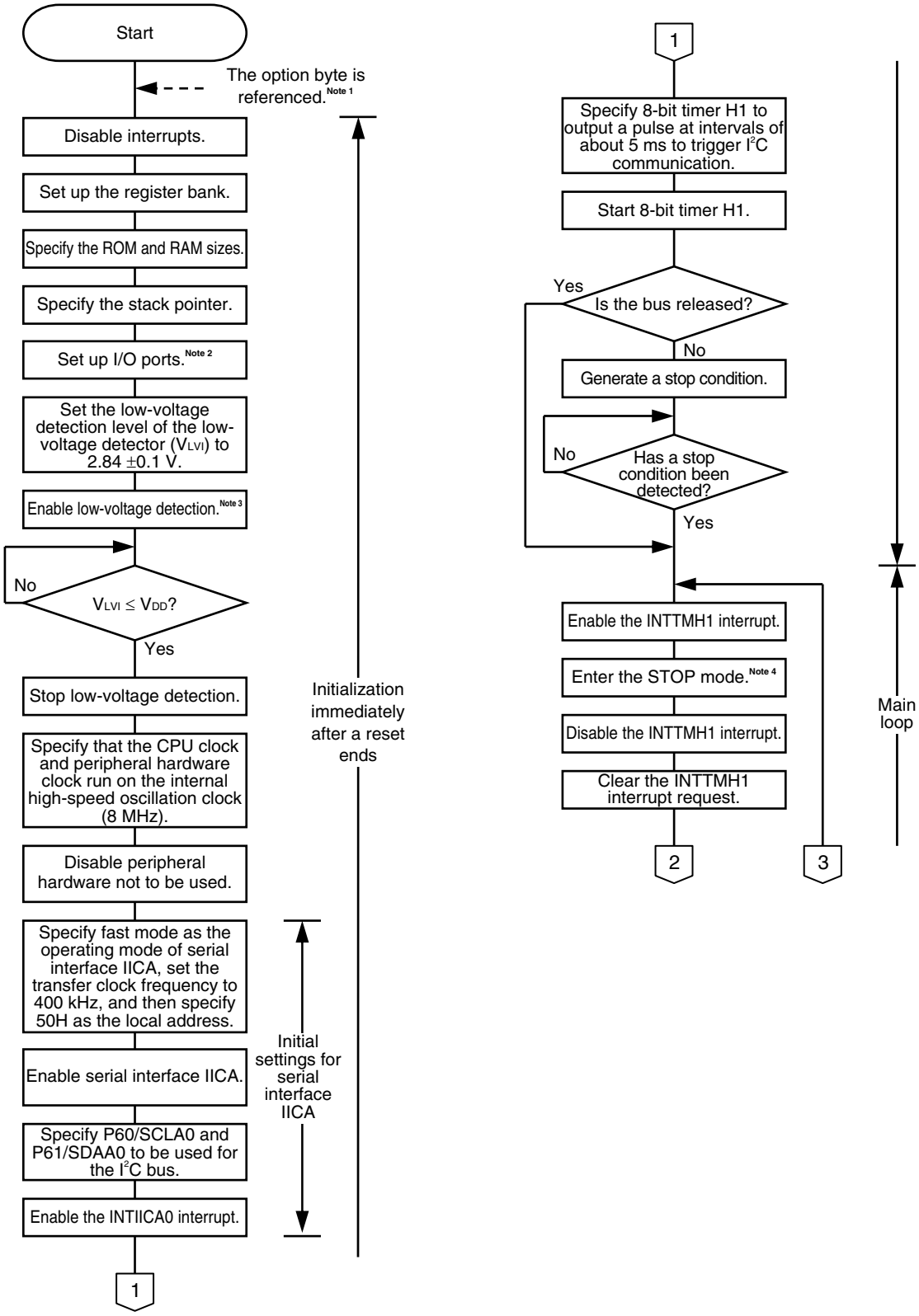
The details are described in the state transition diagram shown below.

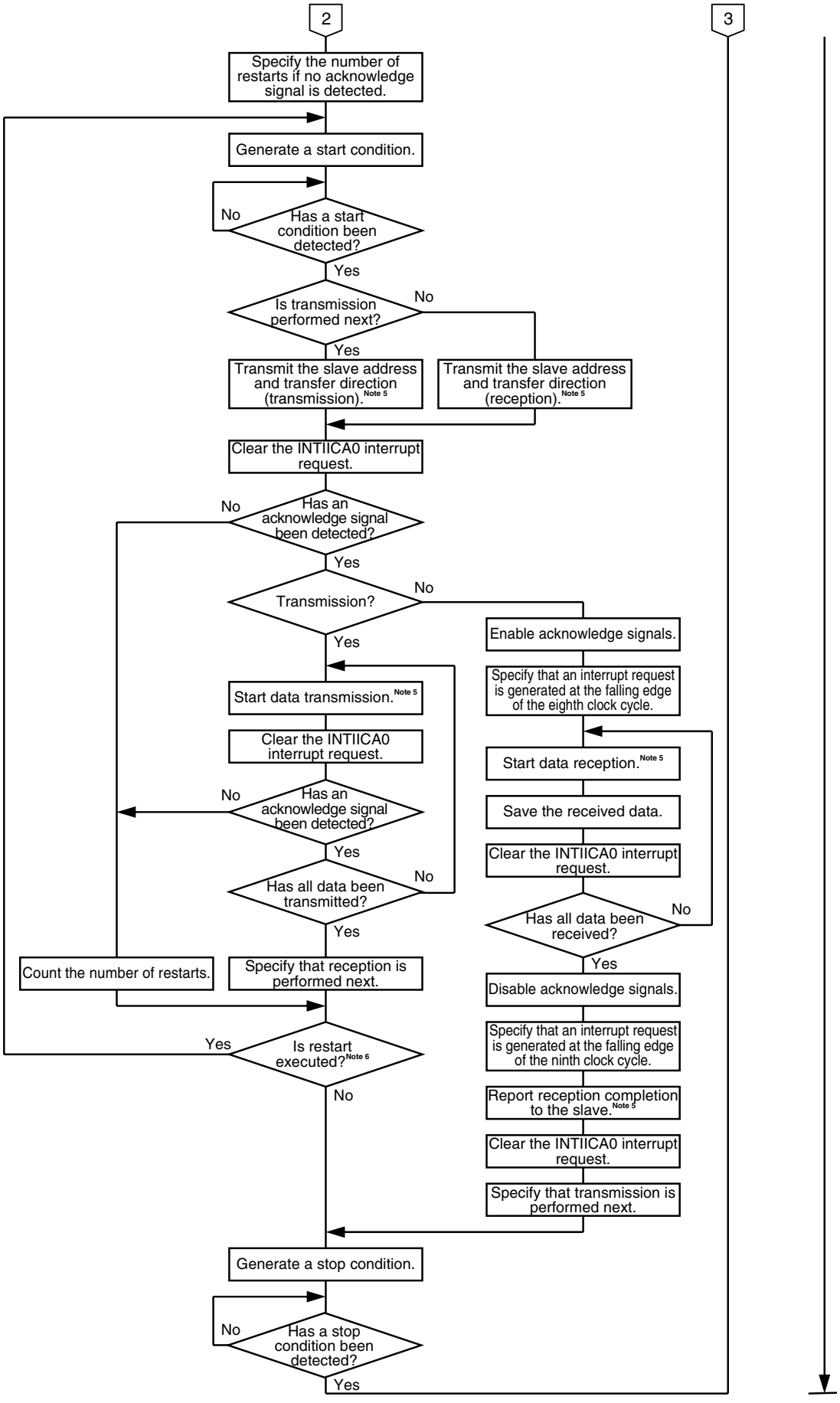


Note This is performed only when detecting an address.

3.4 Flow Charts

The flow charts for the sample program are shown below.




2
3
Specify the number of restarts if no acknowledge signal is detected.
Generate a start condition.
No
Has a start condition been detected?
Yes
Is transmission performed next?
No
Yes
Transmit the slave address and transfer direction (transmission).Note 5
Transmit the slave address and transfer direction (reception).Note 5
Clear the INTIICA0 interrupt request.
No
Has an acknowledge signal been detected?
Yes
Transmission?
No
Yes
Enable acknowledge signals.
Specify that an interrupt request is generated at the falling edge of the eighth clock cycle.
Start data transmission.Note 5
Clear the INTIICA0 interrupt request.
Start data reception.Note 5
No
Has an acknowledge signal been detected?
Save the received data.
Clear the INTIICA0 interrupt request.
Yes
Has all data been transmitted?
No
Has all data been received?
No
Yes
Yes
Count the number of restarts.
Specify that reception is performed next.
Disable acknowledge signals.
Specify that an interrupt request is generated at the falling edge of the ninth clock cycle.
Yes
Is restart executed?Note 6
No
Report reception completion to the slave.Note 5
Clear the INTIICA0 interrupt request.
Specify that transmission is performed next.
Generate a stop condition.
No
Has a stop condition been detected?
Yes

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. P60/SCLA0 and P61/SDAA0 are specified as input ports so that port output does not affect the I^2C bus.

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

4. The STOP mode is exited when the INTTMH1 interrupt is generated at intervals of about 5 ms by using 8-bit timer H1.

5. The HALT mode is entered during communication, and then exited when the INTIICA0 interrupt occurs at the end of communication.

6. Make the system wait about 10 μs before executing a restart.

CHAPTER 4 SETTING METHODS

This chapter describes how to set up serial interface IICA and provides software coding examples.

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 Serial Interface IICA

Serial interface IICA uses the following eight registers.

- IICA control register 0 (IICACTL0)
- IICA flag register 0 (IICAF0)
- IICA control register 1 (IICACTL1)
- IICA low-level width setting register (IICWL)
- IICA high-level width setting register (IICWH)
- Port output mode register 6 (POM6)
- Port mode register 6 (PM6)
- Port register 6 (P6)

[Example of the setup procedure when using serial interface IICA for I^2C master communication]

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

<1> Set bits 0 and 1 (PM60 and PM61) of PM6 to 1 (input mode).[Note]

<2> Set up the transfer clock by using IICWL and IICWH.

<3> Specify the local address by using SVA0.

<4> Specify the conditions for starting I^2C communication by using bit 1 (STCEN) of IICAF0.

<5> Set bit 2 (ACKE0) of IICACTL0 to 1 (to enable acknowledge signals).

<6> Set bit 3 (WTIM0) of IICACTL0 to 1 (to generate an interrupt request at the falling edge of the ninth clock cycle).

<7> Specify the operating mode and operation of the digital filter by using bit 3 (SMC0) and bit 2 (DFC0) of IICACTL1, respectively.

<8> Set bit 7 (IICE0) of IICACTL0 to 1 (to enable the I^2C bus).

<9> Set bits 0 and 1 (POM60 and POM61) of POM6 to 1 (N-ch open-drain output (V_{DD} withstand voltage) mode).

<10> Set bits 0 and 1 (P60 and P61) of P6 to 1 (to output 1).

<11> Clear bits 0 and 1 (PM60 and PM61) of PM6 to 0 (output mode).

<12> Clear the INTIICA0 interrupt request (clear IICAIF0 to 0).

<13> Enable the INTIICA0 interrupt (clear IICAMK0 to 0).

Note P60/SCLA0 and P61/SDAA0 are specified as input ports so that port output does not affect the I^2C bus.

(1) IICA control register 0 (IICACTL0)

This register is used to enable/stop I^2C operations, set wait timing, and set other I^2C operations.

Figure 4-1. Format of IICA Control Register 0 (IICACTL0) (1/4)

IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0

WREL0[Note 1]	Wait cancellation
0	Do not cancel wait
1	Cancel wait. This setting is automatically cleared after wait is canceled.
When WREL0 is set (wait canceled) during the wait period at the ninth clock pulse in the transmission status (TRC0 = 1), the SDAA0 line goes into the high impedance state (TRC0 = 0).	
Condition for clearing (WREL0 = 0)	Condition for setting (WREL0 = 1)
• Automatically cleared after execution • Reset	• Set by instruction

LREL0[Note 1]	Exit from communications
0	Normal operation
1	This exits from the current communications and sets standby mode. This setting is automatically cleared to 0 after being executed. Its uses include cases in which a locally irrelevant extension code has been received. The SCLA0 and SDAA0 lines are set to high impedance. The following flags of IICA control register 0 (IICACTL0) and IICA status register 0 (IICAS0) are cleared to 0. • STT0 • SPT0 • MSTS0 • EXC0 • COI0 • TRC0 • ACKD0 • STD0
The standby mode following exit from communications remains in effect until the following communications entry conditions are met. • After a stop condition is detected, restart is in master mode. • An address match or extension code reception occurs after the start condition.	
Condition for clearing (LREL0 = 0)	Condition for setting (LREL0 = 1)
• Automatically cleared after execution • Reset	• Set by instruction

IICE0	I^2C operation enable
0	Stop operation. Reset the IICA status register 0 (IICAS0)[Note 2]. Stop internal operation.
1	Enable operation.
Be sure to set this bit (1) while the SCLA0 and SDAA0 lines are at high level.	
Condition for clearing (IICE0 = 0)	Condition for setting (IICE0 = 1)
• Cleared by instruction • Reset	• Set by instruction

Notes 1. The signal of this bit is invalid while IICE0 is 0.

2. The IICAS0 register, the STCF and IICBSY bits of the IICAF0 register, and the CLD0 and DAD0 bits of the IICACTL1 register are reset.

Caution The start condition is detected immediately after I^2C is enabled to operate (IICE0 = 1) while the SCLA0 line is at high level and the SDAA0 line is at low level. Immediately after enabling I^2C to operate (IICE0 = 1), set LREL0 (1) by using a 1-bit memory manipulation instruction.

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

Figure 4-1. Format of IICA Control Register 0 (IICACTL0) (2/4)

IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0

ACKE0[Notes 1, 2]	Acknowledgment control
0	Disable acknowledgment.
1	Enable acknowledgment. During the ninth clock period, the SDAA0 line is set to low level.

Condition for clearing (ACKE0 = 0)	Condition for setting (ACKE0 = 1)
• Cleared by instruction • Reset	• Set by instruction

WTIM0[Note 1]	Control of wait and interrupt request generation
0	Interrupt request is generated at the eighth clock's falling edge. Master mode: After output of eight clocks, clock output is set to low level and wait is set. Slave mode: After input of eight clocks, the clock is set to low level and wait is set for master device.
1	Interrupt request is generated at the ninth clock's falling edge. Master mode: After output of nine clocks, clock output is set to low level and wait is set. Slave mode: After input of nine clocks, the clock is set to low level and wait is set for master device.

An interrupt is generated at the falling edge of the ninth clock during address transfer independently of the setting of this bit. The setting of this bit is valid when the address transfer is completed. When in master mode, a wait is inserted at the falling edge of the ninth clock during address transfers. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after an acknowledge ($\overline{ACK}$) is issued. However, when the slave device has received an extension code, a wait is inserted at the falling edge of the eighth clock.

Condition for clearing (WTIM0 = 0)	Condition for setting (WTIM0 = 1)
• Cleared by instruction • Reset	• Set by instruction

SPIE0[Note 1]	Enable/disable generation of interrupt request when stop condition is detected
0	Disable
1	Enable

Condition for clearing (SPIE0 = 0)	Condition for setting (SPIE0 = 1)
• Cleared by instruction • Reset	• Set by instruction

Notes 1. The signal of this bit is invalid while IICE0 is 0. Set this bit during that period.

2. The set value is invalid during address transfer and if the code is not an extension code.
When the device serves as a slave and the addresses match, an acknowledgment is generated regardless of the set value.

Figure 4-1. Format of IICA Control Register 0 (IICACTL0) (3/4)

IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0

STT0[Note]	Start condition trigger
0	Do not generate a start condition.
1	When bus is released (in STOP mode): Generate a start condition (for starting as master). When the SCLA0 line is high level, the SDAA0 line is changed from high level to low level and then the start condition is generated. Next, after the rated amount of time has elapsed, SCLA0 is changed to low level (wait state). When a third party is communicating: • When communication reservation function is enabled (IICRSV = 0) Functions as the start condition reservation flag. When set to 1, automatically generates a start condition after the bus is released. • When communication reservation function is disabled (IICRSV = 1) STCF is set to 1 and information that is set (1) to STT0 is cleared. No start condition is generated. In the wait state (when master device): Generates a restart condition after releasing the wait.
Cautions concerning set timing • For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the wait period when ACKE0 has been cleared to 0 and slave has been notified of final reception. • For master transmission: A start condition cannot be generated normally during the acknowledge period. Set to 1 during the wait period that follows output of the ninth clock. • Cannot be set to 1 at the same time as SPT0. • Setting STT0 to 1 and then setting it again before it is cleared to 0 is prohibited.	
Condition for clearing (STT0 = 0)	Condition for setting (STT0 = 1)
• Cleared by setting STT0 to 1 while communication reservation is prohibited. • Cleared by loss in arbitration • Cleared after start condition is generated by master device • Cleared by LREL0 = 1 (exit from communications) • When IICE0 = 0 (operation stop) • Reset	• Set by instruction

Note The signal of this bit is invalid while IICE0 is 0.

Remarks 1. Bit 1 (STT0) becomes 0 when it is read after data setting.

2. IICRSV: Bit 0 of IICA flag register 0 (IICAF0)

STCF: Bit 7 of IICA flag register 0 (IICAF0)

Figure 4-1. Format of IICA Control Register 0 (IICACTL0) (4/4)

IICE0	LREL0	WREL0	SPIE0	WTIM0	ACKE0	STT0	SPT0

SPT0	Stop condition trigger
0	Do not generate a stop condition.
1	Generate a stop condition (termination of master device's transfer). After the SDAA0 line goes to low level, either set the SCLA0 line to high level or wait until it goes to high level. Next, after the rated amount of time has elapsed, the SDAA0 line changes from low level to high level and a stop condition is generated.

Cautions concerning set timing

- For master reception: Cannot be set to 1 during transfer.
  Can be set to 1 only in the wait period when ACKE0 has been cleared to 0 and slave has been notified of final reception.
- For master transmission: A stop condition cannot be generated normally during the acknowledge period.
  Therefore, set it during the wait period that follows output of the ninth clock.
- Cannot be set to 1 at the same time as STT0.
- SPT0 can be set to 1 only when in master mode[Note].
- When WTIM0 has been cleared to 0, if SPT0 is set to 1 during the wait period that follows output of eight clocks, note that a stop condition will be generated during the high-level period of the ninth clock. WTIM0 should be changed from 0 to 1 during the wait period following the output of eight clocks, and SPT0 should be set to 1 during the wait period that follows the output of the ninth clock.
- Setting SPT0 to 1 and then setting it again before it is cleared to 0 is prohibited.

Condition for clearing (SPT0 = 0)	Condition for setting (SPT0 = 1)
• Cleared by loss in arbitration • Automatically cleared after stop condition is detected • Cleared by LREL0 = 1 (exit from communications) • When IICE0 = 0 (operation stop) • Reset	• Set by instruction

Note Set SPT0 to 1 only in master mode. However, SPT0 must be set to 1 and a stop condition generated before the first stop condition is detected following the switch to the operation enabled status.

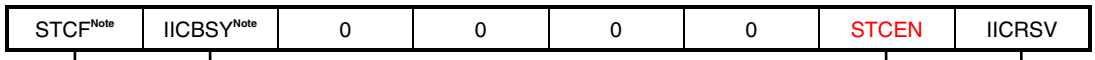
Caution When bit 3 (TRC0) of the IICA status register 0 (IICAS0) is set to 1, WREL0 is set to 1 during the ninth clock and wait is canceled, after which TRC0 is cleared and the SDAA0 line is set to high impedance.

Remark Bit 0 (SPT0) becomes 0 when it is read after data setting.

(2) IICA flag register 0 (IICAF0)

This register sets the operating mode of I^2C and indicates the status of the I^2C bus.

Figure 4-2. Format of IICA Flag Register 0 (IICAF0)



IICRSV	Communication reservation function disable bit
0	Enable communication reservation
1	Disable communication reservation

Condition for clearing (IICRSV = 0)	Condition for setting (IICRSV = 1)
• Cleared by instruction • Reset	• Set by instruction

STCEN	Initial start enable trigger
0	After operation is enabled (IICE0 = 1), enable generation of a start condition upon detection of a stop condition.
1	After operation is enabled (IICE0 = 1), enable generation of a start condition without detecting a stop condition.

Condition for clearing (STCEN = 0)	Condition for setting (STCEN = 1)
• Cleared by instruction • Detection of start condition • Reset	• Set by instruction

IICBSY	I^2C bus status flag
0	Bus release status (communication initial status when STCEN = 1)
1	Bus communication status (communication initial status when STCEN = 0)

Condition for clearing (IICBSY = 0)	Condition for setting (IICBSY = 1)
• Detection of stop condition • When IICE0 = 0 (operation stop) • Reset	• Detection of start condition • Setting of IICE0 when STCEN = 0

STCF	STT0 clear flag
0	Generate start condition
1	Start condition generation unsuccessful: clear STT0 flag

Condition for clearing (STCF = 0)	Condition for setting (STCF = 1)
• Cleared by STT0 = 1 • When IICE0 = 0 (operation stop) • Reset	• Generating start condition unsuccessful and STT0 cleared to 0 when communication reservation is disabled (IICRSV = 1).

Note Bits 7 and 6 are read-only.

Cautions 1. Write to STCEN only when the operation is stopped (IICE0 = 0).

2. As the bus release status (IICBSY = 0) is recognized regardless of the actual bus status when STCEN = 1, when generating the first start condition (STT0 = 1), it is necessary to verify that no third party communications are in progress in order to prevent such communications from being destroyed.

3. Write to IICRSV only when the operation is stopped (IICE0 = 0).

4. Be sure to clear bits 5 to 2 to "0".

Remarks 1. STT0: Bit 1 of IICA control register 0 (IICACTL0)
IICE0: Bit 7 of IICA control register 0 (IICACTL0)

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

(3) IICA control register 1 (IICACTL1)

This register is used to set the operating mode of I^2C and detect the statuses of the SCLA0 and SDAA0 pins.

Figure 4-3. Format of IICA Control Register 1 (IICACTL1)

WUP	0	CLD0[Note 1]	DAD0[Note 1]	SMC0	DFC0	0	0

DFC0	Digital filter operation control
0	Digital filter off
1	Digital filter on
Digital filter can be used only in fast mode. In fast mode, the transfer clock does not vary, regardless of the DFC0 bit being set (1) or cleared (0). The digital filter is used for noise elimination in fast mode.	

SMC0	Operating mode switching
0	Operates in standard mode.
1	Operates in fast mode.

DAD0	Detection of SDAA0 pin level (valid only when IICE0 = 1)
0	The SDAA0 pin was detected at low level.
1	The SDAA0 pin was detected at high level.

Condition for clearing (DAD0 = 0)	Condition for setting (DAD0 = 1)
• When the SDAA0 pin is at low level • When IICE0 = 0 (operation stop) • Reset	• When the SDAA0 pin is at high level

CLD0	Detection of SCLA0 pin level (valid only when IICE0 = 1)
0	The SCLA0 pin was detected at low level.
1	The SCLA0 pin was detected at high level.

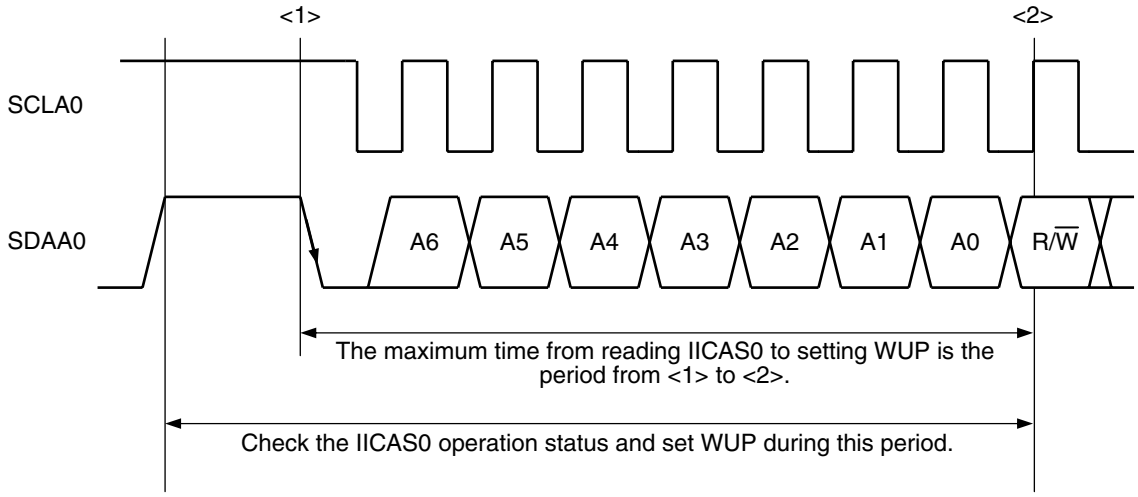
Condition for clearing (CLD0 = 0)	Condition for setting (CLD0 = 1)
• When the SCLA0 pin is at low level • When IICE0 = 0 (operation stop) • Reset	• When the SCLA0 pin is at high level

WUP	Control of address match wakeup
0	Stops operation of address match wakeup function in STOP mode.
1	Enables operation of address match wakeup function in STOP mode.
Clear (0) WUP after the address has matched or an extension code has been received. The subsequent communication can be entered by clearing (0) WUP. (The wait must be released and transmit data must be written after WUP has been cleared (0).) The interrupt timing when the address has matched or when an extension code has been received, while WUP = 1, is identical to the interrupt timing when WUP = 0. (A delay of the difference of sampling by the clock will occur.) Furthermore, when WUP = 1, a stop condition interrupt is not generated even if the SPIE0 bit is set to 1. When WUP = 0 is set by a source other than an interrupt from serial interface IICA, operation as the master device cannot be performed until the subsequent start condition or stop condition is detected. Do not output a start condition by setting (1) the STT0 bit, without waiting for the detection of the subsequent start condition or stop condition.	

Condition for clearing (WUP = 0)	Condition for setting (WUP = 1)
• Cleared by instruction (after address match or extension code reception)	• Set by instruction (when MSTS0, EXC0, and COI0 are “0”, and STD0 also “0” (communication not entered))[Note 2]

Notes 1. Bits 5 and 4 are read-only.

2. The status of IICAS0 must be checked and WUP must be set during the period shown below.



Caution Be sure to clear bits 6, 1, and 0 to "0".

Remarks 1. IICE0: Bit 7 of IICA control register 0 (IICACTL0)

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

(4) IICA low-level width setting register (IICWL), IICA high-level width setting register (IICWH)

The IICA low-level width setting register (IICWL) is used to set the low-level width (t_{LOW}) of the SCLA0 pin signal that is output by serial interface IICA being in master mode.

The IICA high-level width setting register (IICWH) is used to set the high-level width (t_{HIGH}) of the SCLA0 pin signal that is output by serial interface IICA being in master mode.

Figure 4-4. Format of IICA Low-Level Width Setting Register (IICWL)

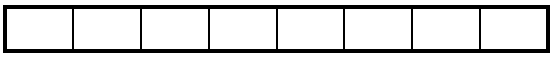
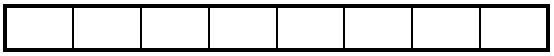


Figure 4-5. Format of IICA High-Level Width Setting Register (IICWH)



The master transfer clock is set up as follows by using the IICWL and IICWH registers:

$$\text{Transfer clock} = \frac{\text{IICWL} + \text{IICWH} + f_{PRS}\,(t_R + t_F)}{f_{PRS}}$$

The ideal values for IICWL and IICWH are shown below.
(The values are rounded to the nearest integer.)

- Fast mode

$$\text{IICWL} = \frac{0.52}{\text{Transfer clock}} \times f_{PRS}$$

$$\text{IICWH} = \left(\frac{0.48}{\text{Transfer clock}} - t_R - t_F\right) \times f_{PRS}$$

- Standard mode

$$\text{IICWL} = \frac{0.47}{\text{Transfer clock}} \times f_{PRS}$$

$$\text{IICWH} = \left(\frac{0.53}{\text{Transfer clock}} - t_R - t_F\right) \times f_{PRS}$$

Caution The transfer clock frequency range that can be specified varies depending on the operating mode.
Standard mode: 0 to 100 kHz
Fast mode: 0 to 400 kHz

Remarks 1. t_F: Falling time of the SDAA0 and SCLA0 signals
t_R: Rising time of the SDAA0 and SCLA0 signals
(For details about t_F and t_R, see the electrical specifications in the 78K0/Kx2-L User's Manual.)
f_{PRS}: Peripheral hardware clock frequency

2. In this sample program, IICWL is set to 11 and IICWH is set to 8.

(5) Port output mode register 6 (POM6)

This register sets the output mode of P60 and P61 in 1-bit units. During I^2C communication, set SCLA0/P60 and SDAA0/P61 to N-ch open drain output (V_{DD} tolerance) mode.

Figure 4-6. Format of Port Output Mode Register 6 (POM6)

0	0	0	0	POM63[Note]	POM62[Note]	POM61	POM60

POM6n	P6n pin output mode selection (n = 1, 0)
0	Normal output (CMOS output) mode
1	N-ch open drain output (V_{DD} tolerance) mode

POM6n	P6n pin output mode selection (n = 3, 2)[Note]
0	Normal output (CMOS output) mode
1	N-ch open drain output (V_{DD} tolerance) mode

Note 78K0/KC2-L only

Caution Be sure to clear the following bits to 0:

78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L: Bits 7 to 2

78K0/KC2-L: Bits 7 to 4

(6) Port register 6 (P6)

This register writes the data to be output from the chip if port 6 is specified to output data.

If using the P60/SCLA0 pin as a clock signal I/O pin and the P61/SDAA0 pin as a serial data I/O pin, set the P60 and P61 output latches to 1.

Figure 4-7. Format of Port Register 6 (P6)

0	0	0	0	P63[Note]	P62[Note]	P61	P60

P61, 60	Output data control (in output mode)	Input data read (in input mode)
0	Output 0.	Input low level.
1	Output 1.	Input high level.

P63, 62[Note]	Output data control (in output mode)	Input data read (in input mode)
0	Output 0.	Input low level.
1	Output 1.	Input high level.

Note 78K0/KC2-L only

Caution Be sure to clear the following bits to 0:

78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L: Bits 7 to 2

78K0/KC2-L: Bits 7 to 4

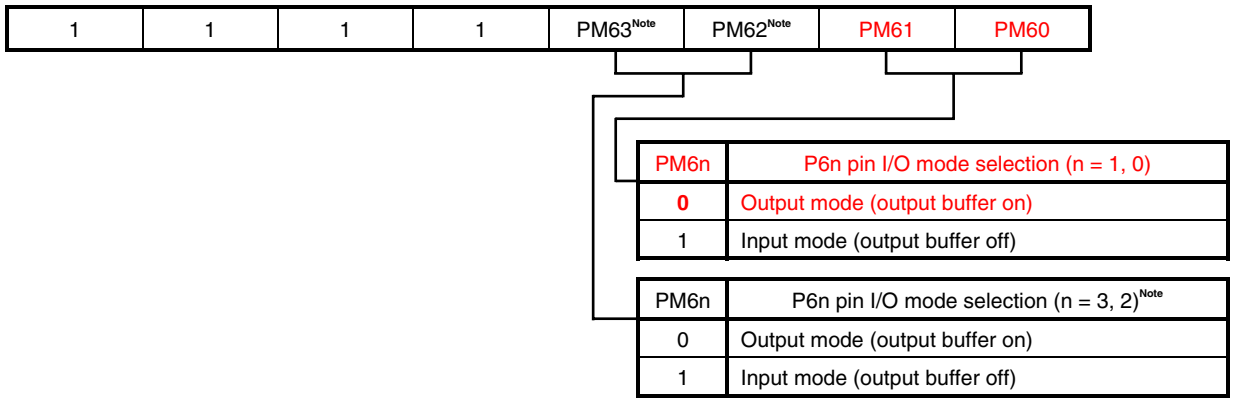
(7) Port mode register 6 (PM6)

This register sets the input/output of port 6 in 1-bit units.

If using the P60/SCLA0 pin as a clock signal I/O pin and the P61/SDAA0 pin as a serial data I/O pin, clear PM60 and PM61 to 0.

Set IICE0 (bit 7 of IICA control register 0 (IICACTL0)) to 1 before setting the output mode because the P60/SCLA0 and P61/SDAA0 pins output a low level (fixed) when IICE0 is 0.

Figure 4-8. Format of Port Mode Register 6 (PM6)



Note 78K0/KC2-L only

Caution Be sure to set the following bits to 1:

78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L: Bits 7 to 2

78K0/KC2-L: Bits 7 to 4

4.2 Software Coding Example

The initialization of serial interface IICA, master transmission, and master reception performed by the 78K0/KC2-L source program are shown below as a software coding example.

(1) Assembly language

<1> Initializing serial interface IICA (common to master transmission and master reception)

```
XMAIN  CSEG   UNIT
IRESET:
···(Omitted)···

      MOV    PM6,   #11110011B          ; Specify P60 and P61 as input ports (to prevent
the I2C bus from being affected)

···(Omitted)···
      MOV    IICWL, #11                 ; Specify the low-level width
      MOV    IICWH, #8                  ; Specify the high-level width

      MOV    SVA0,  #50H                ; Specify the local address

      MOV    IICAF0,#00000000B          ; IICA flag register 0

···(Omitted)···
      MOV    IICACTL0,#00001100B        ; IICA control register 0

···(Omitted)···
      MOV    IICACTL1,#00001100B        ; IICA control register 1

···(Omitted)···
      SET1   IICE0                      ; Enable I2C

      ; Enable I2C bus output
      MOV    POM6,  #00000011B          ; Set P60/SCLA0 and P61/SDAA0 to N-ch open-drain
                                        ; output mode
      MOV    P6,    #00000011B          ; Set the P60/SCLA0 and P61/SDAA0 output latches
to high level
      MOV    PM6,   #11110000B          ; Specify P60/SCLA0 and P61/SDAA0 as output ports

      CLR1   IICAIF0                    ; Clear the INTIICA0 interrupt request
      CLR1   IICAMK0                    ; Enable the INTIICA0 interrupt
```

P60/SCLA0 and P61/SDAA0 are specified as input ports so that port output does not affect the I²C bus.

Set the transfer clock frequency to 400 kHz.

Specify 50H as the local address.

Specify the conditions for starting I²C communication.

Enable acknowledge signals and generate an interrupt request at the falling edge of the ninth clock cycle.

Specify fast mode as the operating mode and enable the digital filter.

Enable the I²C bus.

Specify P60/SCLA0 and P61/SDAA0 to be used for the I²C bus.

Enable the INTIICA0 interrupt.

<2> Master transmission

```
MMAIN_LOOP:

···(Omitted)···
      BT     STCEN, $LMAIN100          ; Has the bus been released? Yes,
      SET1   SPT0                      ; Generate a stop condition
LMAIN050:
      BF     SPD0,  $LMAIN050          ; Has a stop condition been detected? No,

···(Omitted)···
      SET1   STT0                      ; Generate a start condition.
LMAIN160:
      BF     STD0,  $LMAIN160          ; Has a start condition been detected? No,

···(Omitted)···
      MOV    A,     RTRSMOD            ; Read the transfer direction (transmission
(0)/reception (1))
      OR     A,     #CSLVADR           ; Specify an address and the transfer direction
      CALL   !SIICTX                   ; Call transmission processing

···(Omitted)···
      MOVW   HL,    #TTXDATA           ; Specify the start address of the transmission
data table
      MOV    C,     #CDATANUM          ; Specify the number of data units
LMAIN300:
      MOV    A,     [HL]               ; Read the transmission data
      CALL   !SIICTX                   ; Call transmission processing

···(Omitted)···
      SET1   SPT0                      ; Generate a stop condition
LMAIN810:
      BF     SPD0,  $LMAIN810          ; Has a stop condition been detected? No,
```

Check the bus release status.

Generate a start condition.

Transmit the address and transfer direction (transmission).

Transmit data.

Generate a stop condition.

```
SIICTX:
      MOV    IICA,  A                  ; Start transmission

      ; Make the system wait until communication ends
      HALT                             ; Enter the HALT mode (Exit the HALT mode by
generating an INTIICA0 interrupt)
      CLR1   IICAIF0                   ; Clear the INTIICA0 interrupt request

      RET
```

<3> Master reception

```
MMAIN_LOOP:

 ···(Omitted)···
       BT     STCEN, $LMAIN100          ; Has the bus been released? Yes,
       SET1   SPT0                      ; Generate a stop condition
LMAIN050:
       BF     SPD0,  $LMAIN050          ; Has a stop condition been detected? No,

 ···(Omitted)···
       SET1   STT0                      ; Generate a start condition.
LMAIN160:
       BF     STD0,  $LMAIN160          ; Has a start condition been detected? No,

 ···(Omitted)···
       MOV    A,     RTRSMOD            ; Read the transfer direction (transmission
(0)/reception (1))
       OR     A,     #CSLVADR           ; Specify an address and the transfer direction
       CALL   !SIICTX                   ; Call transmission processing

 ···(Omitted)···
       MOVW   HL,    #RRXBUF            ; Specify the start address of the reception
data save area
       MOV    C,     #CDATANUM          ; Specify the number of data units
       SET1   ACKE0                     ; Enable acknowledge signals
       CLR1   WTIM0                     ; Generate an interrupt request at the falling
edge of the eighth clock cycle

LMAIN700:
       ; Start reception
       SET1   WREL0                     ; Make the system exit the wait status
       HALT                             ; Enter the HALT mode (Exit the HALT mode by
generating an INTIICA0 interrupt)
       CLR1   IICAIF0                   ; Clear the INTIICA0 interrupt request

       MOV    A,     IICA               ; Read the received data
       MOV    [HL],  A                  ; Save the received data
       INCW   HL                        ; Go to the next reception data save area
       DBNZ   C,     $LMAIN700          ; Has all reception data been received? No,

       ; When reception ends
       CLR1   ACKE0                     ; Disable acknowledge signals
       SET1   WTIM0                     ; Generate an interrupt request at the falling
edge of the ninth clock cycle
       SET1   WREL0                     ; Make the system exit the wait status
       HALT                             ; Enter the HALT mode (Exit the HALT mode by
generating an INTIICA0 interrupt)
       CLR1   IICAIF0                   ; Clear the INTIICA0 interrupt request

 ···(Omitted)···
       SET1   SPT0                      ; Generate a stop condition
LMAIN810:
       BF     SPD0,  $LMAIN810          ; Has a stop condition been detected? No,
```

```
SIICTX:
       MOV    IICA,  A                  ; Start transmission

       ; Make the system wait until communication ends
       HALT                             ; Enter the HALT mode (Exit the HALT mode by
generating an INTIICA0 interrupt)
       CLR1   IICAIF0                   ; Clear the INTIICA0 interrupt request

       RET
```

Check the bus release status.

Generate a start condition.

Transmit the address and transfer direction (reception).

Start reception.

16 bytes are received.

Report reception completion to the slave.

Generate a stop condition.

(2) C language

<1> Initializing serial interface IICA (common to master transmission and master reception)

```
void hdwinit(void){

  ···(Omitted)···
  PM6  = 0b11110011;        /* Specify P60 and P61 as input ports (to prevent the I2C bus
from being affected) */

  ···(Omitted)···
  IICWL = 11;               /* Specify the low-level width */
  IICWH = 8;                /* Specify the high-level width */

  SVA0 = 0x50;              /* Specify the local address */

  IICAF0 = 0b00000000;      /* IICA flag register 0 */

  ···(Omitted)···
  IICACTL0 = 0b00001100;    /* IICA control register 0 */

  ···(Omitted)···
  IICACTL1 = 0b00001100;    /* IICA control register 1 */

  ···(Omitted)···
  IICE0 = 1;                /* Enable I2C */

  /* Enable I2C bus output */
  POM6 = 0b00000011;        /* Set P60/SCLA0 and P61/SDAA0 to N-ch open-drain */
                            /* output mode                                    */
  P6   = 0b00000011;        /* Set the P60/SCLA0 and P61/SDAA0 output latches to
high level */
  PM6  = 0b11110000;        /* Specify P60/SCLA0 and P61/SDAA0 as output ports */

  IICAIF0    = 0;           /* Clear the INTIICA0 interrupt request */
  IICAMK0    = 0;           /* Enable the INTIICA0 interrupt */
```

P60/SCLA0 and P61/SDAA0 are specified as input ports so that port output does not affect the I²C bus.

Set the transfer clock frequency to 400 kHz.

Specify 50H as the local address.

Specify the conditions for starting I²C communication.

Enable acknowledge signals and generate an interrupt request at the falling edge of the ninth clock cycle.

Specify fast mode as the operating mode and enable the digital filter.

Enable the I²C bus.

Specify P60/SCLA0 and P61/SDAA0 to be used for the I²C bus.

Enable the INTIICA0 interrupt.

<2> Master transmission

```
void main(void)
{

  ···(Omitted)···
  if( !STCEN ){
    SPT0 = 1;                  /* Generate a stop condition */
    while( !SPD0 ){
      NOP();                   /* Make the system wait until a stop condition is detected */
    }
  }

   ···(Omitted)···
     /* Prepare to start communication (generate a start condition) */
     STT0 = 1;                 /* Generate a start condition */
     while( !STD0 ){
        NOP();                 /* Make the system wait until a start condition is detected */
      }

   ···(Omitted)···
        fn_IICTx( CSLAVEADDR );  /* Transmit the address + W (0: data transmission) */

   ···(Omitted)···
           fn_IICTx( aTxData[ucDataCounter] );      /* Transmit data */

   ···(Omitted)···
     SPT0 = 1;                 /* Generate a stop condition */
     while( !SPD0 ){
       NOP();                  /* Make the system wait until a stop condition is detected */
     }
```

Check the bus release status.

Generate a start condition.

Transmit the address and transfer direction (transmission).

Transmit data.

Generate a stop condition.

```
static void fn_IICTx(unsigned char ucTxData)
{
  IICA = ucTxData;             /* Start transmission */

  /* Make the system wait until communication ends */
  HALT();                      /* Enter the HALT mode (Exit the HALT mode by generating an
INTIICA0 interrupt) */
  IICAIF0     = 0;             /* Clear the INTIICA0 interrupt request */
}
```

<3> Master reception

```
void main(void)
{

   ···(Omitted)···
  if( !STCEN ){
    SPT0 = 1;                /* Generate a stop condition */
    while( !SPD0 ){
      NOP();                 /* Make the system wait until a stop condition is detected */
    }
  }

   ···(Omitted)···
     /* Prepare to start communication (generate a start condition) */
     STT0 = 1;               /* Generate a start condition */
     while( !STD0 ){
        NOP();               /* Make the system wait until a start condition is detected */
      }

   ···(Omitted)···
     fn_IICTx( (CSLAVEADDR + 1) );  /* Transmit the address + R (1: data reception) */

   ···(Omitted)···
          ACKE0 = 1;         /* Enable acknowledge signals */
          WTIM0 = 0;         /* Generate an interrupt request at the falling edge of the
eighth clock cycle */

   /* Receive the specified number of bytes */
   for( ucDataCounter = 0; ucDataCounter < 16; ucDataCounter++ ){
        /* Start reception */
      WREL0 = 1;             /* Make the system exit the wait status */
      HALT();                /* Enter the HALT mode (Exit the HALT mode by generating
an INTIICA0 interrupt) */
      IICAIF0 = 0;           /* Clear the INTIICA0 interrupt request */

      ucRxBuffer[ucDataCounter] = IICA;  /* Read the received data */
        }

   /* When reception ends */
          ACKE0 = 0;         /* Disable acknowledge signals */
          WTIM0 = 1;         /* Generate an interrupt request at the falling edge of the
ninth clock cycle */
          WREL0 = 1;         /* Make the system exit the wait status */
          HALT();            /* Enter the HALT mode (Exit the HALT mode by generating an
INTIICA0 interrupt) */
          IICAIF0 = 0;       /* Clear the INTIICA0 interrupt request */

   ···(Omitted)···
    SPT0 = 1;                /* Generate a stop condition */
    while( !SPD0 ){
      NOP();                 /* Make the system wait until a stop condition is detected */
    }
```

```
static void fn_IICTx(unsigned char ucTxData)
{
  IICA = ucTxData;   /* Start transmission */

  /* Make the system wait until communication ends */
  HALT();            /* Enter the HALT mode (Exit the HALT mode by generating an INTIICA0
interrupt) */
  IICAIF0     = 0;   /* Clear the INTIICA0 interrupt request */
}
```

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 (Serial Interface IICA)
;*******************************************************************************
;    Master Communication
;*******************************************************************************
;<<History>>
;    2009.1.--    Release
;*******************************************************************************
;
;<<Overview>>
;
; This sample program presents an example of using serial interface IICA.
; 16 bytes of data are transmitted and received via the I2C bus in master operation.
; Communication starts at intervals of about 5 ms, alternating between transmission
; and reception.
;
;
; <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
; - Checking whether VDD is 2.7 V or more by using the low-voltage detector
; - Specifying that the CPU clock and peripheral hardware clock run on the internal
;    high-speed oscillation clock (8 MHz)
; - Disabling peripheral hardware not to be used
; - Setting up serial interface IICA
;    → Specifying fast mode as the operating mode and setting the transfer clock
frequency to 400 kHz
;    → Specifying 50H as the local address
;    → Specifying that P60/SCLA0 and P61/SDAA0 are used for the I2C bus
;    → Enabling the INTIICA0 interrupt
; - Specifying that 8-bit timer H1 runs in intervals of about 5 ms to trigger I2C
```

```
communication
  ;
  ;
  ; <Communication format>
  ;
  ;   [Transmission]  ST + ADR/W + DT*16 + SP
  ;   [Reception]     ST + ADR/R + DT*16 + SP
  ;
  ;     ST    : Start condition
  ;     SP    : Stop condition
  ;     ADR/W : Slave address + W
  ;     ADR/R : Slave address + R
  ;     DT    : Data
  ;
  ;
  ; <Address and data>
  ;
  ;   Slave address     : A0H
  ;   Transmission data : 16 bytes (00H, 01H, 02H, ... 0DH, 0EH, 0FH)
  ;   Reception data    : 16 bytes (any)
  ;
  ;
  ; <I/O port settings>
  ;  Output: P60, P61
  ;  * 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 ROM data table
  ;
  ;=============================================================================
  XTBL CSEG   AT     0200H
  ; Transmission data table (16 bytes)
  TTXDATA:    DB     00H,01H,02H,03H,04H,05H,06H,07H,08H,09H,0AH,0BH,0CH,0DH,0EH,0FH
  TTXDATAE:                      ; Last address of the transmission data table + 1

  ;=============================================================================
  ;
  ;    Define variables and constants
  ;
  ;=============================================================================
  DRAM DSEG   SADDRP
  RRXBUF:            DS     16           ; Reception data save area (16 bytes)
   CDATANUM   EQU    16           ;  Number  of  data  units  (common  to  transmission  and
reception)
  RTRSMOD:    DS     1            ; Next operation
   CTXMOD            EQU    0            ; Transmission
   CRXMOD            EQU    1            ; Reception

  CSLVADR            EQU    0A0H         ; Slave address
  CRESTRT            EQU    3            ;  Number of restarts if no acknowledge signal is
detected
```

```
;================================================================================
;
;    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,    #00000000B   ; Set the P00 to P02 output latches to low level
    MOV    PM0,   #11111000B   ; Specify P00 to P02 as output ports
                               ; P00 to 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    PM6,   #11110011B   ; Specify P60 and P61 as input ports (to prevent the I2C
bus from being affected)
                                ; Specify P62 and P63 as output ports
       MOV    P6,    #00000000B   ; Set the P60 to P63 output latches to low level
                                ; P60: Use as SCLA0
                                ; P61: Use as SDAA0
                                ; P62 and 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.
;--------------------------------------------------------------------------
     MOV    OSCCTL,#00000000B   ; 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 as an I/O port pin by
specifying 000 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,   #00000000B    ; 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

     MOV    PER0,  #00000000B    ; Control the real-time counter control clock
;                  |+++++++------- Be sure to clear this bit to 0
;                  +-------------- RTCEN:
;                                  [Real-time counter control clock]
;                                   0: Stop supply of control clock
;                                   1: Supply control clock

;-------------------------------------------------------------------------------
;    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 H0
     MOV    TMHMD0,       #00000000B    ; Stop the timer

     ; Real-time counter
     MOV    RTCC0, #00000000B    ; Stop the counter

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

     ; Interrupts
     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 serial interface IICA
  ;---------------------------------------------------------------------------
  ;    - Specify fast mode as the operating mode and set the transfer clock frequency to
400 kHz
  ;    - Specify 50H as the local address
  ;---------------------------------------------------------------------------
      ; Set up the transfer clock
      MOV    IICWL, #11          ; Specify the low-level width
      MOV    IICWH, #8           ; Specify the high-level width

      MOV    SVA0,  #50H         ; Specify the local address

      MOV    IICAF0,#00000000B   ; IICA flag register 0
  ;                  |||||||+------- IICRSV
  ;                  |||||||       [Communication reservation function disable bit]
  ;                  |||||||        0: Enable communication reservation
  ;                  |||||||        1: Disable communication reservation
  ;                  ||||||+-------- STCEN
  ;                  ||||||        [Initial start enable trigger]
  ;                  ||||||         0: After operation is enabled (IICE0 = 1), enable
  ;                  ||||||             generation of a start condition upon detection
  ;                  ||||||             of a stop condition
  ;                  ||||||         1: After operation is enabled (IICE0 = 1), enable
  ;                  ||||||             generation of a start condition without
  ;                  ||||||             detecting a stop condition
  ;                  ||++++--------- Be sure to clear this bit to 0
  ;                  |+-------------- IICBSY <Read only>
  ;                  |              [I2C bus status flag]
  ;                  |               0: Bus release status (communication initial status
when STCEN = 1)
  ;                  |               1: Bus communication status (communication initial
status when STCEN = 0)
  ;                  +--------------- STCF <Read only>
  ;                                  [STT0 clear flag]
  ;                                   0: Generate start condition
  ;                                   1: Start condition generation unsuccessful:
  ;                                       clear STT0 flag

      MOV    IICACTL0,#00001100B ; IICA control register 0
  ;                     |||||||+------ SPT0
  ;                     |||||||    [Stop condition trigger]
```

```
  ;                    |||||||     0: Do not generate a stop condition
  ;                    |||||||     1: Generate a stop condition
  ;                    ||||||+------- STT0
  ;                    ||||||      [Start condition trigger]
  ;                    ||||||       0: Do not generate a start condition
  ;                    ||||||       1: Generate a start condition
  ;                    |||||+-------- ACKE0
  ;                    |||||       [Acknowledgment control]
  ;                    |||||        0: Disable acknowledgment
  ;                    |||||        1: Enable acknowledgment
  ;                    ||||+--------- WTIM0
  ;                    ||||        [Control of wait and interrupt request generation]
  ;                    ||||         0: Interrupt request is generated at the eighth
clock’s falling edge
  ;                    ||||         1: Interrupt request is generated at the ninth
clock’s falling edge
  ;                    |||+---------- SPIE0
  ;                    |||         [Enable/disable generation of interrupt request
  ;                    |||          when stop condition is detected]
  ;                    |||           0: Disable
  ;                    |||           1: Enable
  ;                    ||+----------- WREL0
  ;                    ||          [Wait cancellation]
  ;                    ||           0: Do not cancel wait
  ;                    ||           1: Cancel wait
  ;                    |+------------ LREL0
  ;                    |           [Exit from communications]
  ;                    |            0: Normal operation
  ;                    |            1: This exits from the current communications and
sets standby mode
  ;                   +------------- IICE0
  ;                                [I2C operation enable]
  ;                                 0: Stop operation
  ;                                 1: Enable operation

      MOV    IICACTL1,#00001100B ; IICA control register 1
  ;                    ||||||++------ Be sure to clear this bit to 0
  ;                    |||||+-------- DFC0
  ;                    |||||       [Digital filter operation control]
  ;                    |||||        0: Digital filter off
  ;                    |||||        1: Digital filter on
  ;                    ||||+--------- SMC0
  ;                    ||||        [Operating mode switching]
  ;                    ||||         0: Operates in standard mode
  ;                    ||||         1: Operates in fast mode
  ;                    |||+---------- DAD0
  ;                    |||         [Detection of SDA0 pin level (valid only when IICE =
1)]
```

```
  ;                     |||          0: The SDA0 pin was detected at low level
  ;                     |||          1: The SDA0 pin was detected at high level
  ;                     ||+----------- CLD0
  ;                     ||          [Detection of SCL0 pin level (valid only when IICE =
1)]
  ;                     ||           0: The SCL0 pin was detected at low level
  ;                     ||           1: The SCL0 pin was detected at high level
  ;                     |+------------ Be sure to clear this bit to 0
  ;                     +------------- WUP
  ;                                  [Control of address match wakeup]
  ;                                   0: Stop operation of address match wakeup
  ;                                       function in STOP mode
  ;                                   1: Enable operation of address match wakeup
  ;                                       function in STOP mode

       SET1   IICE0               ; Enable I2C

       ; Enable I2C bus output
       MOV    POM6,  #00000011B   ; Set P60/SCLA0 and P61/SDAA0 to N-ch open-drain
                                  ; output mode
       MOV    P6,    #00000011B   ; Set the P60/SCLA0 and P61/SDAA0 output latches to high
level
       MOV    PM6,   #11110000B   ; Specify P60/SCLA0 and P61/SDAA0 as output ports

       CLR1   IICAIF0             ; Clear the INTIICA0 interrupt request
       CLR1   IICAMK0             ; Enable the INTIICA0 interrupt

  ;---------------------------------------------------------------------------
  ;    Set up 8-bit timer H1
  ;---------------------------------------------------------------------------
       SET1   TMMKH1              ; Disable the INTTMH1 interrupt
       MOV    TMHMD1,       #01110000B   ; Count clock fIL (= 30 kHz)
       MOV    CMP01, #(150-1)     ; Interval of about 5 ms (= 150/fIL)
       CLR1   TMIFH1              ; Clear the INTTMH1 interrupt request
       SET1   TMHE1               ; Start 8-bit timer H1

       BR     MMAIN_LOOP          ; Go to the main loop


  ;*****************************************************************************
  ;
  ;    Main loop
  ;
  ;*****************************************************************************
  MMAIN_LOOP:
       ; Initialize the variables to use
       MOV    RTRSMOD,#CTXMOD     ; Specify that transmission is performed next
```

```
      ; Prepare to start communication (generate a stop condition)
      BT     STCEN, $LMAIN100     ; Has the bus been released? Yes,
      SET1   SPT0                 ; Generate a stop condition
  LMAIN050:
      BF     SPD0,  $LMAIN050     ; Has a stop condition been detected? No,

      ; Communication standby status
  LMAIN100:
      CLR1   TMMKH1               ; Enable the INTTMH1 interrupt
      STOP                        ; Enter the STOP mode (Exit the STOP mode by generating
an INTTMH1 interrupt)
      SET1   TMMKH1               ; Disable the INTTMH1 interrupt
      CLR1   TMIFH1               ; Clear the INTTMH1 interrupt request

  ;---------------------------------------------------------------------------
  ;    Prepare to start I2C communication
  ;---------------------------------------------------------------------------
      MOV    B,     #CRESTRT      ; Specify the number of restarts if no acknowledge
signal is detected

      ; Prepare to start communication (generate a start condition)
  LMAIN150:
      SET1   STT0                 ; Generate a start condition
  LMAIN160:
      BF     STD0,  $LMAIN160     ; Has a start condition been detected? No,

  ;---------------------------------------------------------------------------
  ;    Transmit the address
  ;---------------------------------------------------------------------------
      ; Start communication (transmitting the address and transfer direction)
      MOV    A,     RTRSMOD       ;    Read    the    transfer    direction    (transmission
(0)/reception (1))
      OR     A,     #CSLVADR      ; Specify an address and the transfer direction
      CALL   !SIICTX              ; Call transmission processing
      BF     ACKD0, $LMAIN500     ; Has an acknowledge signal been detected? No,

      BF     TRC0,  $LMAIN600     ; Is data being received? Yes,
  ;---------------------------------------------------------------------------
  ;    Transmit data
  ;---------------------------------------------------------------------------
      MOVW   HL,    #TTXDATA      ; Specify the start address of the transmission data
table
      MOV    C,     #CDATANUM     ; Specify the number of data units
  LMAIN300:
      MOV    A,     [HL]          ; Read the transmission data
      CALL   !SIICTX              ; Call transmission processing
      BF     ACKD0, $LMAIN500     ; Has an acknowledge signal been detected? No,
```

```
      INCW   HL                  ; Go to the next transmission data
      DBNZ   C,     $LMAIN300    ; Has all transmission data been transmitted? No,

      MOV    RTRSMOD,#CRXMOD     ; Perform reception next time
      BR     LMAIN800

  ;---------------------------------------------------------------------------
  ;    Identify restarts
  ;---------------------------------------------------------------------------
  LMAIN500:
      DEC    B                   ; Update the number of restarts
      BZ     $LMAIN800           ; Has a restart ended? Yes,

      ; Make the system wait upon a restart (about 10 us)
      MOV    C,     #10          ; Specify the number of counts
  LMAIN550:
      NOP                        ;
      DBNZ   C,     $LMAIN550    ; Has the wait period ended? No,
      BR     LMAIN150            ; Go to preparing to start communication

  ;---------------------------------------------------------------------------
  ;    Receive data
  ;---------------------------------------------------------------------------
  LMAIN600:
      MOVW   HL,    #RRXBUF      ; Specify the start address of the reception data save
area
      MOV    C,     #CDATANUM    ; Specify the number of data units
      SET1   ACKE0               ; Enable acknowledge signals
      CLR1   WTIM0               ; Generate an interrupt request at the falling edge of
the eighth clock cycle

  LMAIN700:
      ; Start reception
      SET1   WREL0               ; Make the system exit the wait status
      HALT                       ; Enter the HALT mode (Exit the HALT mode by generating
an INTIICA0 interrupt)
      CLR1   IICAIF0             ; Clear the INTIICA0 interrupt request

      MOV    A,     IICA         ; Read the received data
      MOV    [HL],  A            ; Save the received data
      INCW   HL                  ; Go to the next reception data save area
      DBNZ   C,     $LMAIN700    ; Has all reception data been received? No,

      ; When reception ends
      CLR1   ACKE0               ; Disable acknowledge signals
      SET1   WTIM0               ; Generate an interrupt request at the falling edge of
the ninth clock cycle
      SET1   WREL0               ; Make the system exit the wait status
```

```
      HALT                    ; Enter the HALT mode (Exit the HALT mode by generating
an INTIICA0 interrupt)
      CLR1   IICAIF0          ; Clear the INTIICA0 interrupt request

      MOV    RTRSMOD,#CTXMOD  ; Perform transmission next time

  ;-------------------------------------------------------------------------------
  ;    Generate a stop condition
  ;-------------------------------------------------------------------------------
  LMAIN800:
      SET1   SPT0             ; Generate a stop condition
  LMAIN810:
      BF     SPD0,  $LMAIN810 ; Has a stop condition been detected? No,

      BR     LMAIN100

  ;*******************************************************************************
  ;
  ;    Transmission
  ;
  ;-------------------------------------------------------------------------------
  ;    [I N]  A      : 8-bit data to transmit
  ;    [OUT]  -
  ;*******************************************************************************
  SIICTX:
      MOV    IICA,  A         ; Start transmission

      ; Make the system wait until communication ends
      HALT                    ; Enter the HALT mode (Exit the HALT mode by generating
an INTIICA0 interrupt)
      CLR1   IICAIF0          ; Clear the INTIICA0 interrupt request

      RET
  end
```

● main.c (C language version)

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

  NEC Electronics     78K0/KC2-L Series

************************************************************************************
  78K0/KC2-L Series   Sample Program (Serial Interface IICA)
************************************************************************************
  Master Communication
************************************************************************************
<<History>>
  2009.1.-- Release
************************************************************************************

<<Overview>>

This sample program presents an example of using serial interface IICA.
16 bytes of data are transmitted and received via the I2C bus in master operation.
Communication starts at intervals of about 5 ms, alternating between transmission
and reception.


 <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
 - Checking whether VDD is 2.7 V or more by using the low-voltage detector
 - Specifying that the CPU clock and peripheral hardware clock run on the internal
    high-speed oscillation clock (8 MHz)
 - Disabling peripheral hardware not to be used
 - Setting up serial interface IICA
    → Specifying fast mode as the operating mode and setting the transfer clock
frequency to 400 kHz
    → Specifying 50H as the local address
    → Specifying that P60/SCLA0 and P61/SDAA0 are used for the I2C bus
    → Enabling the INTIICA0 interrupt
 - Specifying that 8-bit timer H1 runs in intervals of about 5 ms to trigger I2C
communication


 <Communication format>
```

```
   [Transmission]  ST + ADR/W + DT*16 + SP
   [Reception]     ST + ADR/R + DT*16 + SP

     ST    : Start condition
     SP    : Stop condition
     ADR/W : Slave address + W
     ADR/R : Slave address + R
     DT    : Data


 <Address and data>

   Slave address     : A0H
   Transmission data : 16 bytes (00H, 01H, 02H, ... 0DH, 0EH, 0FH)
   Reception data    : 16 bytes (any)


 <I/O port settings>
  Output: P60, P61
  * 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  HALT            /* HALT instructions can be described at the C source level */
#pragma  STOP            /* STOP instructions can be described at the C source level */

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

  Declare function prototypes

=======================================================================*/
static void fn_IICTx(unsigned char ucTxData);   /* Transmission */

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

  Define the ROM data table

=======================================================================*/
```

```
  static const unsigned char aTxData[16]=
  { /* Transmission data (16 bytes) */
    0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0A,0x0B,0x0C,0x0D,0x0E,0x0F
  };

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

    Define variables and constants

  ====================================================================================*/
  static unsigned char ucRxBuffer[16];      /* Reception data save area */
  static unsigned char ucTransmissionMode;  /*   Next   operation   (0:   transmission,   1:
reception) */
  static unsigned char ucRestartCounter;    /* Number of times to execute a restart */
   #define CRESTART 3                    /* Number of restarts if no acknowledge signal is
detected */
  static unsigned char ucDataCounter;          /*  Number  of  transmission  and  reception
data units */

  #define CSLAVEADDR 0xA0   /* Slave address */


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

    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     = 0b00000000;/* Set the P00 to P02 output latches to low level */
    PM0    = 0b11111000;/* Specify P00 to P02 as output ports */
                        /* P00 to 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
  -------------------------------------------------------------------------*/
    PM6    = 0b11110011;/* Specify P60 and P61 as input ports (to prevent the I2C bus from
being affected) */
                        /* Specify P62 and P63 as output ports */
    P6     = 0b00000000;/* Set the P60 to P63 output latches to low level */
                        /* P60: Use as SCLA0 */
```

```
                        /* P61: Use as SDAA0 */
                        /* P62 and 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.
  -------------------------------------------------------------------------*/
    OSCCTL = 0b00000000; /* 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 as an I/O port pin by specifying 000
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    = 0b00000000;/* 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 */

   PER0   = 0b00000000; /* Control the real-time counter control clock */
   /*         |+++++++---- Be sure to clear this bit to 0 */
   /*         +----------- RTCEN: */
   /*                    [Real-time counter control clock] */
   /*                     0: Stop supply of control clock */
   /*                     1: Supply control clock */

  /*-------------------------------------------------------------------------
    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 H0 */
    TMHMD0 = 0b00000000; /* Stop the timer */

    /* Real-time counter */
    RTCC0  = 0b00000000; /* Stop the counter */

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

    /* Interrupts */
    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 serial interface IICA
  -------------------------------------------------------------------------
    - Specify fast mode as the operating mode and set the transfer clock frequency to 400
kHz
    - Specify 50H as the local address
  -------------------------------------------------------------------------*/
    /* Set up the transfer clock */
    IICWL  = 11;        /* Specify the low-level width */
```

```
    IICWH  = 8;         /* Specify the high-level width */

    SVA0   = 0x50;      /* Specify the local address */

    IICAF0 = 0b00000000;/* IICA flag register 0 */
    /*         |||||||+------ IICRSV                                          */
    /*         |||||||      [Communication reservation function disable bit]  */
    /*         |||||||       0: Enable communication reservation             */
    /*         |||||||       1: Disable communication reservation            */
    /*         ||||||+------- STCEN                                           */
    /*         ||||||       [Initial start enable trigger]                   */
    /*         ||||||        0: After operation is enabled (IICE0 = 1), enable */
    /*         ||||||           generation of a start condition upon detection */
    /*         ||||||           of a stop condition                          */
    /*         ||||||        1: After operation is enabled (IICE0 = 1), enable */
    /*         ||||||           generation of a start condition without      */
    /*         ||||||           detecting a stop condition                   */
    /*         ||++++-------- Be sure to clear this bit to 0                */
    /*         |+------------ IICBSY <Read only>                             */
    /*         |            [I2C bus status flag]                            */
    /*         |             0: Bus release status (communication initial status when STCEN
= 1) */
    /*         |             1: Bus communication status (communication initial status when
STCEN = 0) */
    /*         +------------- STCF <Read only>                                */
    /*                      [STT0 clear flag]                                */
    /*                       0: Generate start condition                     */
    /*                       1: Start condition generation unsuccessful:     */
    /*                          clear STT0 flag                              */

    IICACTL0 = 0b00001100;  /* IICA control register 0 */
    /*           |||||||+----- SPT0                                          */
    /*           |||||||    [Stop condition trigger]                         */
    /*           |||||||     0: Do not generate a stop condition            */
    /*           |||||||     1: Generate a stop condition                   */
    /*           ||||||+------ STT0                                          */
    /*           ||||||     [Start condition trigger]                       */
    /*           ||||||      0: Do not generate a start condition           */
    /*           ||||||      1: Generate a start condition                  */
    /*           |||||+------- ACKE0                                         */
    /*           |||||      [Acknowledgment control]                        */
    /*           |||||       0: Disable acknowledgment                      */
    /*           |||||       1: Enable acknowledgment                       */
    /*           ||||+-------- WTIM0                                         */
    /*           ||||       [Control of wait and interrupt request generation] */
    /*           ||||        0:  Interrupt  request  is  generated  at  the  eighth  clock's
falling edge */
    /*           ||||        1: Interrupt request is generated at the ninth clock's falling
```

```
edge */
    /*           |||+---------- SPIE0                                              */
    /*           |||        [Enable/disable generation of interrupt request      */
    /*           |||         when stop condition is detected]                     */
    /*           |||          0: Disable                                          */
    /*           |||          1: Enable                                           */
    /*           ||+---------- WREL0                                              */
    /*           ||         [Wait cancellation]                                  */
    /*           ||          0: Do not cancel wait                               */
    /*           ||          1: Cancel wait                                      */
    /*           |+----------- LREL0                                             */
    /*           |          [Exit from communications]                           */
    /*           |           0: Normal operation                                 */
    /*           |           1: This exits from the current communications and sets standby
mode   */
    /*           +------------ IICE0                                             */
    /*                      [I2C operation enable]                               */
    /*                       0: Stop operation                                   */
    /*                       1: Enable operation                                 */

    IICACTL1 = 0b00001100;  /* IICA control register 1 */
    /*           ||||||++----- Be sure to clear this bit to 0                    */
    /*           |||||+------- DFC0                                              */
    /*           |||||      [Digital filter operation control]                   */
    /*           |||||       0: Digital filter off                               */
    /*           |||||       1: Digital filter on                                */
    /*           ||||+-------- SMC0                                              */
    /*           ||||       [Operating mode switching]                          */
    /*           ||||        0: Operates in standard mode                       */
    /*           ||||        1: Operates in fast mode                           */
    /*           |||+--------- DAD0                                              */
    /*           |||        [Detection of SDA0 pin level (valid only when IICE = 1)] */
    /*           |||         0: The SDA0 pin was detected at low level          */
    /*           |||         1: The SDA0 pin was detected at high level         */
    /*           ||+---------- CLD0                                              */
    /*           ||         [Detection of SCL0 pin level (valid only when IICE = 1)] */
    /*           ||          0: The SCL0 pin was detected at low level          */
    /*           ||          1: The SCL0 pin was detected at high level         */
    /*           |+----------- Be sure to clear this bit to 0                    */
    /*           +------------ WUP                                               */
    /*                      [Control of address match wakeup]                    */
    /*                       0: Stop operation of address match wakeup          */
    /*                          function in STOP mode                           */
    /*                       1: Enable operation of address match wakeup        */
    /*                          function in STOP mode                           */

    IICE0  = 1;         /* Enable I2C */
```

```
  /* Enable I2C bus output */
  POM6   = 0b00000011;/* Set P60/SCLA0 and P61/SDAA0 to N-ch open-drain */
                      /* output mode                                    */
  P6     = 0b00000011;/* Set the P60/SCLA0 and P61/SDAA0 output latches to high level */
  PM6    = 0b11110000;/* Specify P60/SCLA0 and P61/SDAA0 as output ports */

  IICAIF0   = 0;      /* Clear the INTIICA0 interrupt request */
  IICAMK0   = 0;      /* Enable the INTIICA0 interrupt */

/*-------------------------------------------------------------------------
  Set up 8-bit timer H1
-------------------------------------------------------------------------*/
  TMMKH1 = 1;         /* Disable the INTTMH1 interrupt */
  TMHMD1 = 0b01110000;/* Count clock fIL (= 30 kHz) */
  CMP01  = (150-1);   /* Interval of about 5 ms (= 150/fIL) */
  TMIFH1 = 0;         /* Clear the INTTMH1 interrupt request */
  TMHE1  = 1;         /* Start 8-bit timer H1 */

}


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

  Main loop

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

  /* Initialize the variables to use */
  ucTransmissionMode = 0;    /* Specify that transmission is performed next */

  /* If the bus is not released */
  if( !STCEN ){
     SPT0 = 1;        /* Generate a stop condition */
     while( !SPD0 ){
        NOP();        /* Make the system wait until a stop condition is detected */
     }
  }

  while (1){
     /* Communication standby status */
     TMMKH1 = 0;  /* Enable the INTTMH1 interrupt */
     STOP();      /* Enter the STOP mode (Exit the STOP mode by generating an INTTMH1
interrupt) */
     TMMKH1 = 1;  /* Disable the INTTMH1 interrupt */
     TMIFH1 = 0;  /* Clear the INTTMH1 interrupt request */
```

```
      /* Restart loop */
      /*   * This loop is exited because no restart is necessary if transmission and
reception have been completed */
      for( ucRestartCounter = 0; ucRestartCounter < CRESTART; ucRestartCounter++){
      /*-------------------------------------------------------------------------
         Prepare to start I2C communication
      -------------------------------------------------------------------------*/
         /* Prepare to start communication (generate a start condition) */
         STT0 = 1;        /* Generate a start condition */
         while( !STD0 ){
            NOP();        /* Make the system wait until a start condition is detected */
         }

      /*-------------------------------------------------------------------------
         Transmit the address
      -------------------------------------------------------------------------*/
         /* If transmission is performed next */
         if( !ucTransmissionMode ){
            fn_IICTx( CSLAVEADDR );       /*   Transmit   the   address   +   W   (0:   data
transmission) */
         }
         /* If reception is performed next */
         else{
            fn_IICTx( (CSLAVEADDR + 1) ); /*   Transmit   the   address   +   R   (1:   data
reception) */
         }

         /* If an acknowledge signal is detected */
         if( ACKD0 ){
            /* During transmission */
            if( TRC0 ){
            /*-----------------------------------------------------------------
               Transmit data
            -----------------------------------------------------------------*/
               /* Transmit the specified number of bytes */
               for( ucDataCounter = 0; ucDataCounter < 16; ucDataCounter++ ){
                  fn_IICTx( aTxData[ucDataCounter] );  /* Transmit data */

                  /* If an acknowledge signal is not detected */
                  if( !ACKD0 ){
                     break; /* Suspend transmission */
                  }
               }
            }
            /* During reception */
            else{
            /*-----------------------------------------------------------------
```

```
                Receive data
            ------------------------------------------------------------*/
                ACKE0 = 1;    /* Enable acknowledge signals */
                WTIM0 = 0;    /* Generate an interrupt request at the falling edge of the
eighth clock cycle */

                /* Receive the specified number of bytes */
                for( ucDataCounter = 0; ucDataCounter < 16; ucDataCounter++ ){
                    /* Start reception */
                    WREL0 = 1;    /* Make the system exit the wait status */
                    HALT();       /* Enter the HALT mode (Exit the HALT mode by generating
an INTIICA0 interrupt) */
                    IICAIF0 = 0;  /* Clear the INTIICA0 interrupt request */

                    ucRxBuffer[ucDataCounter] = IICA;/* Read the received data */
                }

                /* When reception ends */
                ACKE0 = 0;    /* Disable acknowledge signals */
                WTIM0 = 1;    /* Generate an interrupt request at the falling edge of the
ninth clock cycle */
                WREL0 = 1;    /* Make the system exit the wait status */
                HALT();       /* Enter the HALT mode (Exit the HALT mode by generating an
INTIICA0 interrupt) */
                IICAIF0 = 0;  /* Clear the INTIICA0 interrupt request */
            }
        }

        /* End transmission and reception */
        /*  * Exit the restart loop and proceed to generating a stop condition */
        if( ucDataCounter >= 16 ){
              ucTransmissionMode ^= 1;   /*      Switch      the      next      operation
(transmission/reception) */
              break;                 /* Go to generating a stop condition  */
        }
        /* Restart */
        else{
            /* Make the system wait upon a restart (about 10 us) */
            for( ucCounter = 0; ucCounter < 2; ucCounter++ ){
                NOP();
            }
        }
    }

  /*-----------------------------------------------------------------
      Generate a stop condition
  -----------------------------------------------------------------*/
      SPT0 = 1;        /* Generate a stop condition */
```

```
      while( !SPD0 ){
          NOP();       /* Make the system wait until a stop condition is detected */
      }
    }
  }

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

    Transmission

  ------------------------------------------------------------------------
    [I N]  ucTxData     : 8-bit data to transmit
    [OUT]  -
  ************************************************************************/
  static void fn_IICTx(unsigned char ucTxData)
  {
    IICA   = ucTxData;      /* Start transmission */

    /* Make the system wait until communication ends */
    HALT();                 /* Enter the HALT mode (Exit the HALT mode by generating an
INTIICA0 interrupt) */
       IICAIF0   = 0;          /* Clear the INTIICA0 interrupt request */
    }
```

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	August 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