Application Note

78K0/Kx2-L

Sample Program (Initial Settings)

LED Lighting Switch Control

This document summarizes the initial settings for the sample program and describes basic initial settings for the microcontroller. In the sample program, the two switch inputs and lighting of the three LEDs are controlled, after the basic initial settings for the microcontroller, such as selecting the clock frequency or I/O port, have been performed.

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

CONTENTS

CHAPTER 1 OVERVIEW **3**
CHAPTER 2 CIRCUIT DIAGRAM **4**
2.1 Circuit Diagram 4
2.2 Used Devices Other than Microcontroller 4
CHAPTER 3 SOFTWARE **5**
3.1 Included Files 5
3.2 Internal Peripheral Functions to Be Used 6
3.3 Initial Settings and Operation Overview 6
3.4 Flow Chart 7
CHAPTER 4 SETTING METHODS **8**
4.1 Setting Up Option Byte 9
4.2 Setting Up Vector Table 15
4.3 Specifying ROM and RAM Sizes 16
4.4 Setting Up Stack Pointer 17
4.5 Setting Up and Controlling Watchdog Timer 18
4.6 Setting Up Clock 19
4.7 Setting Up Ports 27
4.8 Main Processing 34
CHAPTER 5 RELATED DOCUMENTS **36**
APPENDIX A PROGRAM LIST **37**
APPENDIX B USING 78K0/KC2-L 44-PIN PRODUCTS **54**
APPENDIX C REVISION HISTORY **55**

Document No. U19685EJ1V0AN00 (1st edition)
Date Published September 2009 N

CHAPTER 1 OVERVIEW

In this sample program, the basic initial settings for the 78K0/Kx2-L microcontroller, such as setting up the option byte, selecting the clock frequency, and setting up I/O ports are performed. In the main processing operation after completion of the initial settings, the lighting of the three LEDs is controlled by using the two switch inputs.

(1) Primary initial settings

<Option byte settings>

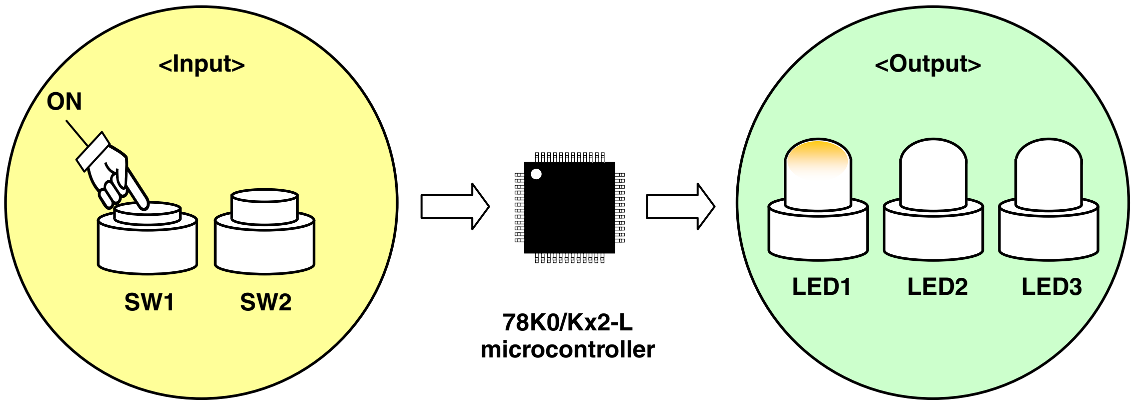
- Allowing the internal low-speed oscillator to be programmed to stop
- Disabling the watchdog timer
- Setting the internal high-speed oscillation clock frequency to 8 MHz
- Disabling LVI from being started by default

<Settings during initialization immediately after a reset ends>

- Specifying the ROM and RAM sizes
- Specifying that the CPU clock run on the internal high-speed oscillation clock (4 MHz)
- Stopping the internal low-speed oscillator
- Setting up I/O ports
- Disabling peripheral hardware not to be used

(2) Main processing operation

Lighting of the LEDs (LED1, LED2, LED3) is controlled by detecting switch inputs (SW1, SW2) with the 78K0/Kx2-L microcontroller.



Switch Input		LED Output		
SW1	SW2	LED1	LED2	LED3
OFF	OFF	OFF	OFF	OFF
ON	OFF	ON	OFF	OFF
OFF	ON	OFF	ON	OFF
ON	ON	OFF	OFF	ON

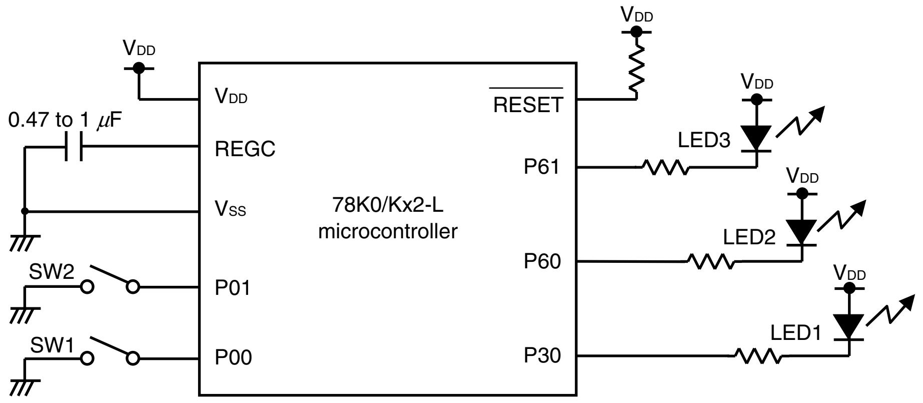
Caution For cautions when using the device, refer to the 78K0/Kx2-L User’s Manual.

CHAPTER 2 CIRCUIT DIAGRAM

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

2.1 Circuit Diagram

A circuit diagram is shown below.



Cautions 1. Use the microcontroller at a voltage in the range of 1.8 V ≤ V_{DD} ≤ 5.5 V.

2. Connect the AV_{REF} pin directly to V_{DD}.

3. Connect the AV_{SS} pin directly to GND (only for the 78K0/KC2-L and 78K0/KB2-L microcontrollers).

4. Connect REGC to V_{SS} via a capacitor (0.47 to 1 μF).

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

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 V_{DD} or V_{SS} via a resistor.

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

2.2 Used Devices Other than Microcontroller

The following devices are used in addition to the microcontroller:

(1) Switches (SW1, SW2)

These switches are used as inputs to control the lighting of the LEDs.

(2) LEDs (LED1, LED2, LED3)

The LEDs are used as outputs corresponding to switch inputs.

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 a flow chart.

3.1 Included Files

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

File Name	Description	Compressed (*.zip) File Included	
		ZIP	PM 32
main.asm (Assembly language version)	Source file for hardware initialization processing and main processing of microcontroller	●Note	●Note
main.c (C language version)			
op.asm	Assembler source file for setting the option byte (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_Init.prw	Work space file for integrated development environment PM+		●
Kx2-L_Init.prj	Project file for integrated development environment PM+		●

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

Remark ZIP : Only the source file is included.

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

3.2 Internal Peripheral Functions to Be Used

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

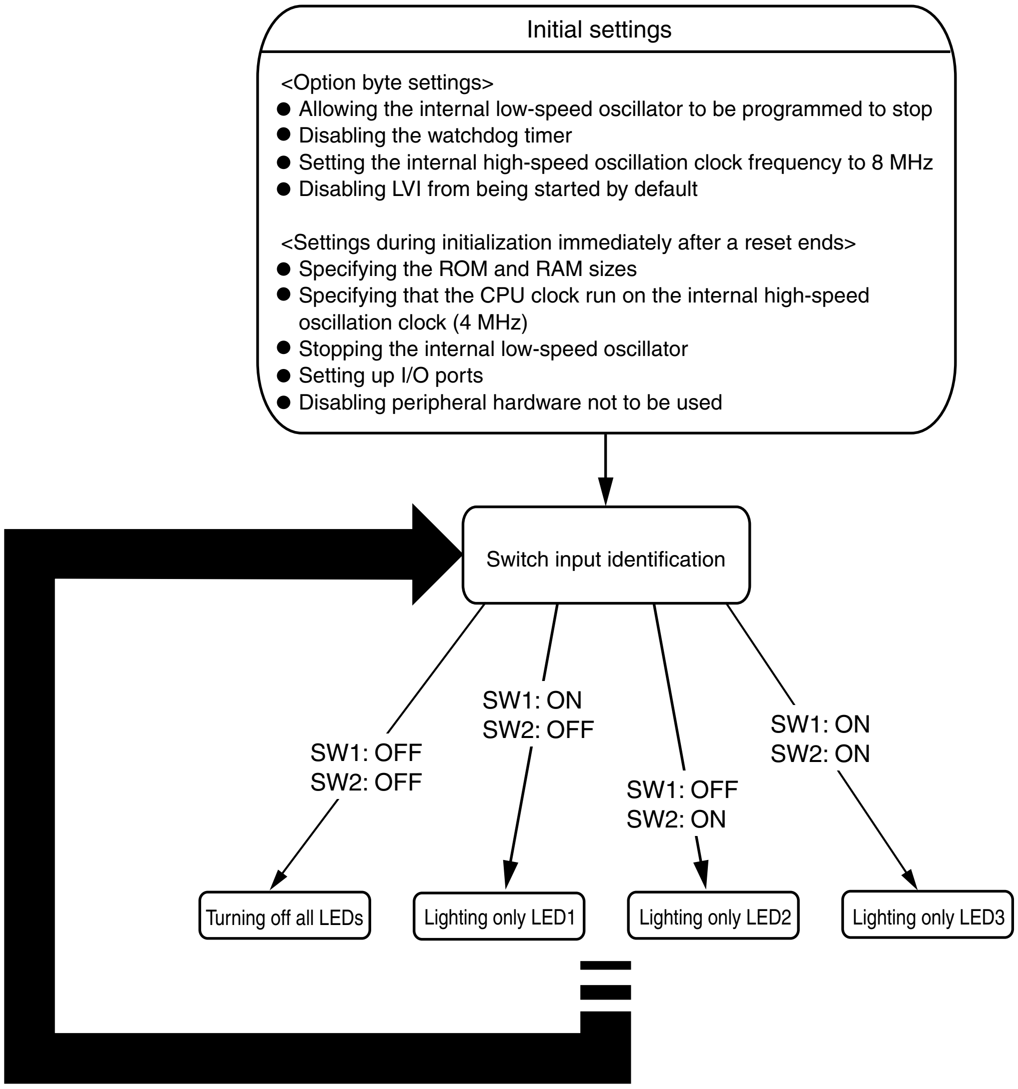
- Input ports (for switch inputs): P00, P01
- Output ports (for lighting LEDs): P30, P60, P61

3.3 Initial Settings and Operation Overview

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

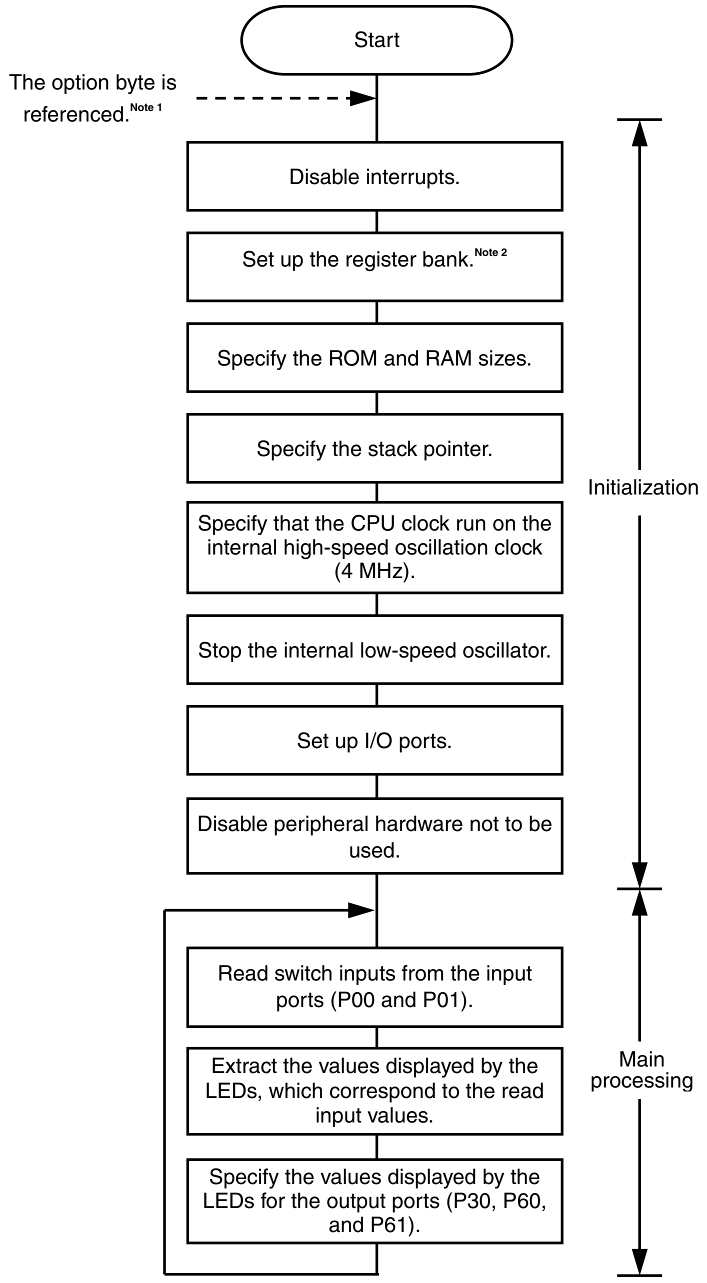
After completion of the initial settings, the lighting of the three LEDs (LED1, LED2, LED3) is controlled in accordance with the combination of the two switch inputs (SW1, SW2).

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


Initial settings
<Option byte settings>
● Allowing the internal low-speed oscillator to be programmed to stop
● Disabling the watchdog timer
● Setting the internal high-speed oscillation clock frequency to 8 MHz
● Disabling LVI from being started by default
<Settings during initialization immediately after a reset ends>
● Specifying the ROM and RAM sizes
● Specifying that the CPU clock run on the internal high-speed oscillation clock (4 MHz)
● Stopping the internal low-speed oscillator
● Setting up I/O ports
● Disabling peripheral hardware not to be used
Switch input identification
SW1: OFF SW2: OFF
SW1: ON SW2: OFF
SW1: OFF SW2: ON
SW1: ON SW2: ON
Turning off all LEDs
Lighting only LED1
Lighting only LED2
Lighting only LED3

3.4 Flow Chart

A flow chart for the sample program is shown below.


Start
The option byte is referenced.Note 1
Disable interrupts.
Set up the register bank.Note 2
Specify the ROM and RAM sizes.
Specify the stack pointer.
Specify that the CPU clock run on the internal high-speed oscillation clock (4 MHz).
Stop the internal low-speed oscillator.
Set up I/O ports.
Disable peripheral hardware not to be used.
Read switch inputs from the input ports (P00 and P01).
Extract the values displayed by the LEDs, which correspond to the read input values.
Specify the values displayed by the LEDs for the output ports (P30, P60, and P61).
Initialization
Main processing

Notes 1. The option byte is automatically referenced by the microcontroller immediately after a reset ends. In this sample program, the following settings are specified using the option byte:
- Allowing the internal low-speed oscillator to be programmed to stop
- Disabling the watchdog timer
- Setting the internal high-speed oscillation clock frequency to 8 MHz
- Disabling LVI from being started by default

2. The general-purpose registers of the 78K0/Kx2-L Series microcontrollers are configured in four register banks so that the registers used for normal processing and those used when an interrupt occurs can be changed on a bank basis in order to create an efficient program. In this sample program, only register bank 0 is used.

CHAPTER 4 SETTING METHODS

This chapter describes how to set up the option byte, vector table, watchdog timer, and I/O ports, how to specify the ROM and RAM sizes, stack pointer, and clock frequency, and provides details about the main processing.

To execute a program written in C, another program that performs ROMization to integrate the former program into the system and starts a user-created program (main function) is required. The latter program is called a startup routine. In general, a startup routine is the first program that runs after the microcontroller is reset (initialized). It initially sets up the hardware such as the CPU, memory, and I/O ports and specifies the initial settings for running the main processing routine. In general, the startup routine, the main routine, and then subroutines are executed, and interrupts are serviced.

In the C version of this sample program, clock settings and initial settings for peripheral hardware are specified using the hdwinit function, after which the main function is executed. Therefore, the main processing is included in the main function. In the assembly language version, the microcontroller is reset (initialized), a program is executed from the IRESET address written at address 0000H in the vector table, clock settings and initial settings for peripheral hardware are specified as by the hdwinit function in the C version, and then the main processing begins.

In this sample program, the peripheral hardware units that are not to be used are disabled. To use them, set up the corresponding registers in accordance with the purpose for using them and their functions.

For details about the startup routine, refer to the chapter about the startup routine in the CC78K0 Operation User's Manual.

For how to set register, 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 Option Byte

The option byte must be set. The following items are set with the option byte.

(1) Internal low-speed oscillator oscillation control

(2) Watchdog timer interval time setting

(3) Watchdog timer counter operation setting

(4) Watchdog timer window open period setting

(5) LVI default start operation control

(6) Internal high-speed oscillation clock frequency selection

(7) On-chip debug operation control

Figure 4-1. Format of Option Byte (Address: 0080H/1080H)

Address: 0080H/1080H[Note]

0	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	LSROSC

LSROSC	Internal low-speed oscillator operation
0	Can be stopped by software (stopped when 1 is written to bit 1 (LSRSTOP) of the RCM register)
1	Cannot be stopped (not stopped even if 1 is written to the LSRSTOP bit)

WDCS2	WDCS1	WDCS0	Watchdog timer overflow time
0	0	0	$2^7/f_{IL}$ (3.88 ms)
0	0	1	$2^8/f_{IL}$ (7.76 ms)
0	1	0	$2^9/f_{IL}$ (15.52 ms)
0	1	1	$2^{10}/f_{IL}$ (31.03 ms)
1	0	0	$2^{12}/f_{IL}$ (124.12 ms)
1	0	1	$2^{14}/f_{IL}$ (496.48 ms)
1	1	0	$2^{15}/f_{IL}$ (992.97 ms)
1	**1**	**1**	$2^{17}/f_{IL}$ (3.97 s)

WDTON	Operation control of watchdog timer counter/illegal access detection
0	Counter operation disabled (counting stopped after reset), illegal access detection operation disabled
1	Counter operation enabled (counting started after reset), illegal access detection operation enabled

WINDOW1	WINDOW0	Watchdog timer window open period
0	0	25%
0	1	50%
1	0	75%
1	**1**	100%

Note Set a value that is the same as that of 0080H to 1080H because 0080H and 1080H are switched during the boot swap operation.

Cautions 1. The combination of WDCS2 = WDCS1 = WDCS0 = 0 and WINDOW1 = WINDOW0 = 0 is prohibited.

2. The watchdog timer continues its operation during self programming and EEPROM emulation of the flash memory. During processing, the interrupt acknowledge time is delayed. Set the overflow time and window size taking this delay into consideration.

3. If LSROSC = 0 (oscillation can be stopped by software), the count clock is not supplied to the watchdog timer in the HALT and STOP modes, regardless of the setting of bit 0 (LSRSTOP) of the internal oscillation mode register (RCM).
When 8-bit timer H1 runs on the internal low-speed oscillation clock, the count clock is supplied to 8-bit timer H1 even in the HALT/STOP mode.

4. Be sure to clear bit 7 to 0.

Remarks 1. f_{IL}: Internal low-speed oscillation clock frequency

2. ( ): f_{IL} = 33 kHz (MAX.)

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

Figure 4-2. Format of Option Byte (Address: 0081H/1081H)

Address: 0081H/1081H[Notes 1, 2]

0	**0**	**0**	**0**	**0**	**0**	**0**	LVISTART

LVISTART	LVI default start operation control
0	LVI is OFF by default after a reset or upon power application (LVI default start function stopped)
1	LVI is ON by default after a reset or upon power application (LVI default start function enabled)

Notes 1. LVISTART can only be written by using a dedicated flash memory programmer. It cannot be set during self programming or boot swap operation during self programming. However, because 0081H and 1081H are switched during the boot swap operation, set a value that is the same as that of 0081H to 1081H.

2. To change the setting for the LVI default start, set the value to 0081H again after batch erasure (chip erasure) of the flash memory. The setting cannot be changed after the memory of the specified block is erased.

Caution Be sure to clear bits 7 to 1 to “0”.

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

Figure 4-3. Format of Option Byte (Address: 0082H/1082H)

Address: 0082H/1082H[Note]

0	**0**	**0**	**0**	**0**	**0**	**0**	R4M8MSEL

R4M8MSEL	Internal high-speed oscillation clock frequency selection
0	8 MHz (TYP.)
1	4 MHz (TYP.)

Note Set a value that is the same as that of 0082H to 1082H because 0082H and 1082H are switched during the boot swap operation.

Caution Be sure to clear bits 7 to 1 to “0”.

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

Figure 4-4. Format of Option Byte (Address: 0083H/1083H)

Address: 0083H/1083H[Note]

0	0	0	OCDCK STOP	1	1	OCDPSEL	OCDONB

OCDONB	Transition of on-chip debug mode
0	Not shifting to on-chip debug mode immediately after a reset ends
1	Shifting to on-chip debug mode immediately after a reset ends

OCDPSEL	Pin selection used during on-chip debugging
0	TOOLC1/P31, TOOLD1/P32
1	TOOLC0/X1, TOOLD0/X2

OCDCK STOP	Internal high-speed oscillator operation control after execution of the STOP instruction in on-chip debug mode
0	Internal high-speed oscillator operation continues. (However, internal high-speed oscillation clock is not supplied to CPU and peripheral hardware.)
1	Internal high-speed oscillator operation stops.

Note Set a value that is the same as that of 0083H to 1083H because 0083H and 1083H are switched during the boot swap operation.

Caution Be sure to clear bits 7 to 5 to "0" and set bits 3 and 2 to "1".

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

Figure 4-5. Format of Option Byte (Address: 0084H/1084H)

Address: 0084H/1084H[Note]

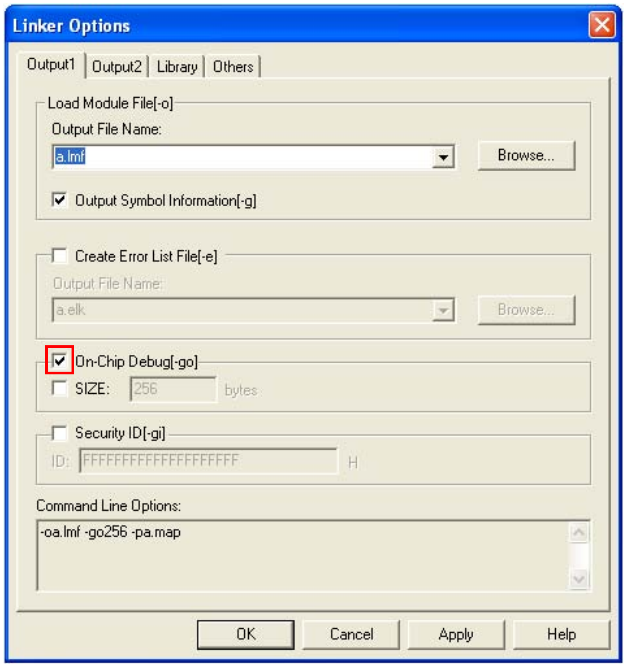
0	**0**	**0**	**0**	**0**	**0**	OCDEN1	OCDEN0

OCDEN1	OCDEN0	On-chip debug operation control
0	0	Operation disabled
0	1	Setting prohibited
1	**0**	Operation enabled. Does not erase data of the flash memory in case authentication of the on-chip debug security ID fails.
1	1	Operation enabled. Erases data of the flash memory in case authentication of the on-chip debug security ID fails.

Note Set a value that is the same as that of 0084H to 1084H because 0084H and 1084H are switched during the boot swap operation.

Cautions 1. Be sure to clear bits 7 to 2 to "0".

2. To perform on-chip debugging, the linker option must be set up in addition to the option byte. To perform on-chip debugging, select [Linker Options] in the PM+ [Tool] menu, and then select [On-Chip Debug [-go]]. For details about on-chip debugging, refer to CHAPTER 26 ON-CHIP DEBUG FUNCTION in the 78K0/Kx2-L User's Manual.


Linker Options

Output1 | Output2 | Library | Others

Load Module File[-o]
Output File Name: a.lmf
[✓] Output Symbol Information[-g]

[ ] Create Error List File[-e]
Output File Name: a.elk

[✓] On-Chip Debug[-go]
[ ] SIZE: 256 bytes

[ ] Security ID[-gi]
ID: FFFFFFFFFFFFFFFFFFFF H

Command Line Options:
-oa.lmf -go256 -pa.map

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

The values specified for the option byte, above, are as follows in the program.

```
TOPTIONB      CSEG   AT     0080H
       DB     01101110B
       DB     00000000B
       DB     00000000B
       DB     00011111B
       DB     00000010B
```

To use C language, prepare an assembly language source file (file name: “*.asm (*: arbitrary)”) such as the one shown below, specify it as the project source file, and build it with other source files (main.c).

```
TOPTIONB      CSEG   AT     0080H
       DB     01101110B
       DB     00000000B
       DB     00000000B
       DB     00011111B
       DB     00000010B
END
```

[Column] What are CSEG (Code Segment), DSEG (Data Segment), and BSEG (Bit Segment)?
CSEG, DSEG, and BSEG are pseudo instructions which indicate where generated codes of instructions, data, or the like are to be allocated. Instructions and data which are described after such pseudo instructions have been issued are allocated in the ROM area with a CSEG pseudo instruction, in the RAM area with a DSEG pseudo instruction, and in the saddr area in RAM with a BSEG pseudo instruction.
For example, to allocate the option byte setting content to addresses starting from 0080H in the internal ROM (flash memory), first, the CSEG pseudo instruction and AT attribute are used to specify 0080H as the address. Next, the DB pseudo instruction is used to define values that are to be set to addresses following 0080H, which are then described in the program coded in assembly language.
The DB and DW pseudo instructions can be used only in a ROM area specified with the CSEG pseudo instruction. Descriptions of the DB or DW pseudo instructions in a RAM area specified with the DSEG or BSEG pseudo instruction will not cause errors, but must not be used. In this case, an object is generated and debug operation can be performed, since with MINICUBE2 (on-chip debug emulator) or SM+ (system simulator), coded instructions and data are expanded to the RAM area. With an actual device, however, operation is disabled since these cannot be expanded to the RAM area.
For details of the CSEG, DSEG, and BSEG pseudo instructions, refer to the RA78K0 Language User’s Manual.

4.2 Setting Up Vector Table

In the vector table area, the program start address, which is used when branching occurs due to the generation of resets and various interrupt requests, is stored. In this sample program, interrupts are not serviced, so only the reset vector which is used during reset start is set.

This setting is required when coding in assembly language. When coding in C language, this setting is not required.

[Setting example] Setting up only the reset vector to be used when starting a reset (same as in the sample program settings)

Address

Function name

```
      XVECT1     CSEG   AT     0000H
<1>         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
<2>         DW     IINIT                ;000AH INTP2

      • • • (Omitted) • • •
            DW     IINIT                ;003AH INTP10
            DW     IINIT                ;003CH INTP11
            DW     IINIT                ;003EH BRK

      • • • (Omitted) • • •
      ;*******************************************************************************
      ;
      ;      Servicing interrupts by using unnecessary interrupt sources
      ;
      ;*******************************************************************************
<3>   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
```

Immediately after the reset ends, the program starts from the address (`RESET_START` at <1>, above) specified using the reset vector.

In this sample program, vector table addresses except 0000H are not used. `IINIT` is specified for all remaining vector table addresses (<2> above). If these settings are specified, even if an interrupt occurs, the processing branches to `IINIT` (<3> above), and then returns from the interrupt without performing processing, assuming the interrupt to be unnecessary.

[Column] What are #pragma directives?

#pragma directives are preprocessing instructions which are used in the C language and are coded at the beginning of source programs.

The following are major #pragma directives.

• #pragma sfr:	Operations related to the SFR area can be specified at the C source level.
• #pragma ei:	The EI instruction can be specified at the C source level.
• #pragma di:	The DI instruction can be specified at the C source level.
• #pragma nop:	The NOP instruction can be specified at the C source level. (The clock can be advanced without operating the CPU.)
• #pragma interrupt:	Interrupt functions can be specified at the C source level.

For details about the #pragma directives, refer to the chapter regarding expansion functions, in the CC78K0 Language User's Manual.

4.3 Specifying ROM and RAM Sizes

The ROM and internal high-speed RAM capacities of a 78K0/Kx2-L Series microcontroller vary depending on the model. Therefore, the memory size switching register (IMS) must be set up during initialization.

The default project device settings (device file settings) are as follows:

78K0/KY2-L: μPD78F0557
78K0/KA2-L: μPD78F0567
78K0/KB2-L: μPD78F0578
78K0/KC2-L: μPD78F0588

The values specified for the IMS register for each model are written in the sample source. Therefore, change the values according to the microcontroller to use.

[Example] Using the μPD78F0586

• Assembly language

Delete ;.

Add ;

```
;-------------------------------------------------------------------------------
;       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
```

To use the μPD78F0586, add a semicolon (;) before `MOV` on the line under `Setting when using uPD78F0583 or uPD78F0588` to disable the line by commenting it out. Next, delete the semicolon (;) before `MOV` on the line under `Setting when using uPD78F0581 or uPD78F0586` to enable the line. The IMS settings can be changed to those for the μPD78F0586.

• C language

Delete /* and */.

Add /* and */.

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

To use the μPD78F0586, add /* before and */ after `IMS = 0xC8;` on the line under `/* Setting when using uPD78F0583 or uPD78F0588 */` to disable the line by commenting it out. Next, delete /* before and */ after `IMS = 0x42;` on the line under `/* Setting when using uPD78F0581 or uPD78F0586 */` to enable the line. The IMS settings can be changed to those for the μPD78F0586.

4.4 Setting Up Stack Pointer

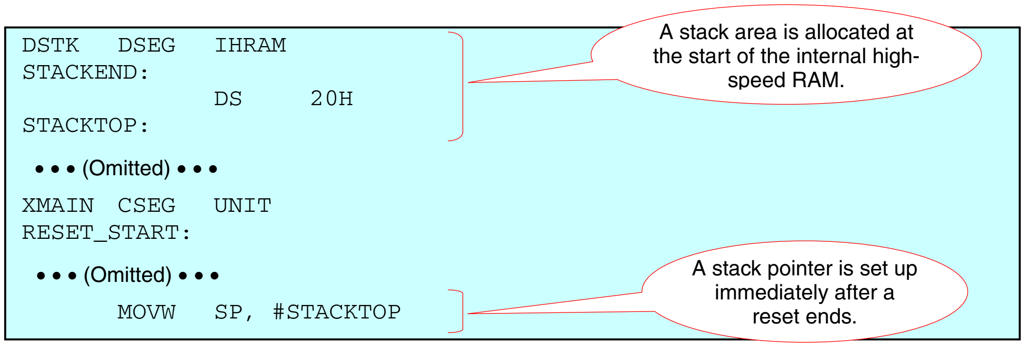
A stack area is a memory area in which data, such as of program counters, register values, and PSW (program status word) is temporarily stored. A stack area can be specified only in the internal high-speed RAM. The start address of this stack area is specified using a stack pointer to allocate the stack area.

A stack area is used when the following instructions are executed or interrupts occur.

• PUSH, CALL, CALLT, CALLF interrupt:	Allocating data to a stack area
• POP, RET, RETI:	Restoring data from a stack area

A stack area must be allocated when coding in assembly language. When coding in C language, this setting is not required, because a stack area is automatically allocated in the startup routine.

[Example] Using the first 32 bytes in the internal high-speed RAM as the stack area
(Same as in the sample program settings)



```
DSTK   DSEG   IHRAM
STACKEND:
              DS      20H
STACKTOP:

 • • • (Omitted) • • •

XMAIN  CSEG   UNIT
RESET_START:

 • • • (Omitted) • • •

       MOVW   SP, #STACKTOP
```

A stack area is allocated at the start of the internal high-speed RAM.
A stack pointer is set up immediately after a reset ends.

By writing the above code, the first 32 bytes in the internal high-speed RAM can be allocated as the stack area.

The start address in the internal high-speed RAM varies depending on the device. The stack is allocated to the area of the following addresses:

78K0/KY2-L: 0FD80H to 0FD9FH
78K0/KA2-L: 0FD00H to 0FD1FH
78K0/KB2-L: 0FC00H to 0FC1FH
78K0/KC2-L: 0FB00H to 0FB1FH

In this sample program, the start of the internal high-speed RAM is specified without writing an absolute address by using the DSEG pseudo instruction IHRAM[Note].

Note For details, refer to the RA78K0 Language User's Manual.

4.5 Setting Up and Controlling Watchdog Timer

The watchdog timer is set up using the option byte. For details, refer to **4.1 Setting Up Option Byte**.

When using the watchdog timer (when WDTON is 1), the watchdog timer is controlled using the watchdog timer enable register (WDTE). The watchdog timer counter is cleared and then starts counting again when ACH is written to WDTE. WDTE is set to 9AH[Note] by generating a reset signal.

Note The WDTE reset value varies depending on the value specified for WDTON of the option byte (0080H).

WDTON Setting	WDTE Reset Value
0 (Watchdog timer count operation disabled)	1AH
1 (Watchdog timer count operation enabled)	9AH

Cautions 1. If a value other than ACH is written to WDTE, an internal reset signal is generated. If the source clock to the watchdog timer is stopped, however, an internal reset signal is generated when the source clock to the watchdog timer resumes operation.

2. If a 1-bit memory manipulation instruction is executed for WDTE, an internal reset signal is generated. If the source clock to the watchdog timer is stopped, however, an internal reset signal is generated when the source clock to the watchdog timer resumes operation.

3. The value read from WDTE is 9AH/1AH (this differs from the written value (ACH)).

[Column] Binary-value description

To describe a binary value, append "B" or "Y" after the binary value in assembly language, or append "0b" or "0B" before the binary value in C language.

[Column] hdwinit function and main function

To create a program in C language, the hdwinit function is called to initialize peripheral devices (SFR) immediately after the CPU is reset. Initial settings, such as setting up the I/O ports and selecting the clock frequency are therefore basically included in the hdwinit function.

The main function is called after calling the hdwinit function, so main processing is included in the main function.

Do not call the hdwinit function from the main function. In this case, the hdwinit function is executed twice and the watchdog timer setting, which is only allowed to be specified once is executed twice. As a result, an internal reset signal is generated during the second execution disabling the program to advance from the initial setting.

For details, refer to the CC78K0 Language User's Manual and Processing to be executed first under Programming on the NEC Electronics FAQ Web page.

4.6 Setting Up Clock

The CPU clock signal (f_{CPU}) and the clock signal supplied to the peripheral hardware (f_{PRS}) are generated by dividing the frequency of the main system clock signal (f_{XP}).

(1) Selecting the clock operation mode

Select the clock operation mode by using the clock operation mode select register (OSCCTL).

Figure 4-6. Format of Clock Operation Mode Select Register (OSCCTL)

OSCCTL

EXCLK	OSCSEL	EXCLKS[Note]	OSCSELS[Note]	**0**	RSWOSC	AMPHXT	**0**

RSWOSC	AMPHXT	XT1 oscillator oscillation mode selection
0	0	Low power consumption oscillation (default)
0	1	Normal oscillation
1	x	Ultra-low power consumption oscillation

EXCLK	OSCSEL	High-speed system clock pin operation mode	P121/X1 pin	P122/X2/EXCLK pin
0	**0**	Input port mode	Input port	
0	1	X1 oscillation mode	Crystal/ceramic resonator connection	
1	0	Input port mode	Input port	
1	1	External clock input mode	Input port	External clock input

Note EXCLKS and OSCSELS can be used only for a device in which the subsystem clock is used (78K0/KC2-L). For details, refer to (3) Specifying the operating mode of the subsystem clock pin.

Cautions 1. To change the value of EXCLK and OSCSEL, be sure to confirm that bit 7 (MSTOP) of the main OSC control register (MOC) is 1 (the X1 oscillator stops or the external clock from the EXCLK pin is disabled).

2. Be sure to clear the following bits to 0:

78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L:	**Bits 5 to 0**
78K0/KC2-L:	**Bits 3 and 0**

3. The XT1 oscillator is a circuit with low amplification in order to achieve low-power consumption. Note the following points when designing the circuit.

- **Make the wiring between the XT1 and XT2 pins and the resonators as short as possible, and minimize the parasitic capacitance and wiring resistance. Note this particularly when the ultra-low power consumption oscillation (RSWOSC = 1) is selected.**
- **Place a ground pattern that has the same potential as V_{SS} as much as possible near the XT1 oscillator.**
- **Be sure that the signal lines between the XT1 and XT2 pins, and the resonators do not cross with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**

4. For details about these cautions, refer to CHAPTER 5 CLOCK GENERATOR in the 78K0/Kx2-L User's Manual.

Remarks 1. x: don't care

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

(2) Selecting the CPU clock (f_{CPU}) and specifying the division ratio

Select the CPU clock (f_{CPU}) and specify the division ratio by using the processor clock control register (PCC).

Figure 4-7. Format of Processor Clock Control Register (PCC)

PCC

0	XTSTART[Note 1]	CLS[Note 2]	CSS[Note 2]	0	PCC2	PCC1	PCC0

CSS	PCC2	PCC1	PCC0	CPU clock (f_{CPU}) selection
0	0	0	0	f_{XP}
0	0	0	1	$f_{XP}/2$
0	0	1	0	$f_{XP}/2^2$
0	0	1	1	$f_{XP}/2^3$
0	1	0	0	$f_{XP}/2^4$
1	0	0	0	$f_{SUB}/2$
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
Other than the above				Setting prohibited

CLS	CPU clock status
0	Main system clock
1	Subsystem clock

Notes 1. XTSTART can be used only for a device in which the subsystem clock is used (78K0/KC2-L). For details, refer to (3) Specifying the operating mode of the subsystem clock pin.

2. CLS and CSS can be used only for a device in which the subsystem clock is used (78K0/KC2-L).

Cautions 1. Be sure to clear the following bits to 0:

78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L:	**Bits 7 to 3**
78K0/KC2-L:	**Bits 7 and 3**

2. The peripheral hardware clock (f_{PRS}) frequency is not divided when the division ratio of the PCC is set.

3. Bit 5 is read-only.

Remarks 1. f_{XP}: Main system clock frequency

2. f_{SUB}: Subsystem clock frequency

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

The fastest instruction can be executed in 2 clocks of the CPU clock in the 78K0/Kx2-L microcontrollers. Therefore, the relationship between the CPU clock (f_{CPU}) and the minimum instruction execution time is as follows:

CPU Clock (f_{CPU})	Minimum Instruction Execution Time: $2/f_{CPU}$			
	Main System Clock			Subsystem Clock[Note 2]
	High-Speed System Clock[Note 1]	Internal High-Speed Oscillation Clock[Note 1]		
	10 MHz Operation	8 MHz (TYP.) Operation	4 MHz (TYP.) Operation	32.768 kHz Operation
f_{XP}	0.2 μs (TYP.)	0.25 μs (TYP.)	0.5 μs (TYP.)	–
$f_{XP}/2$	0.4 μs (TYP.)	0.5 μs (TYP.)	1.0 μs (TYP.)	–
$f_{XP}/2^2$	0.8 μs (TYP.)	1.0 μs (TYP.)	2.0 μs (TYP.)	–
$f_{XP}/2^3$	1.6 μs (TYP.)	2.0 μs (TYP.)	4.0 μs (TYP.)	–
$f_{XP}/2^4$	3.2 μs (TYP.)	4.0 μs (TYP.)	8.0 μs (TYP.)	–
$f_{SUB}/2$	–	–		122.1 μs

Notes 1. The main clock mode register (MCM) is used to set the main system clock supplied to CPU clock (high-speed system clock/internal high-speed oscillation clock).

2. This can be used only for a device in which the subsystem clock is used (78K0/KC2-L).

(3) Specifying the operating mode of the subsystem clock pin[Note]

The operating mode of the subsystem clock pin can be specified using bit 6 (XTSTART) of the processor clock control register (PCC) and bits 5 and 4 (EXCLKS, OSCSELS) of the clock operation mode select register (OSCCTL).

Note This setting can be specified only for a device in which the subsystem clock is used (78K0/KC2-L).

The following table shows the relationship between the values of XTSTART, EXCLKS, and OSCSELS and the operating mode of the subsystem clock pin.

PCC	OSCCTL		Subsystem Clock Pin Operating Mode	P123/XT1 Pin	P124/XT2/EXCLKS Pin
Bit 5	Bit 5	Bit 4			
XTSTART	EXCLKS	OSCSELS			
0	**0**	**0**	Input port mode	Input port	
0	0	1	XT1 oscillation mode	Crystal resonator connection	
0	1	0	Input port mode	Input port	
0	1	1	External clock input mode	Input port	External clock input
1	x	x	XT1 oscillation mode	Crystal resonator connection	

Caution Confirm that bit 5 (CLS) of the processor clock control register (PCC) is 0 (the CPU runs on the main system clock) when changing the current values of XTSTART, EXCLKS, and OSCSELS.

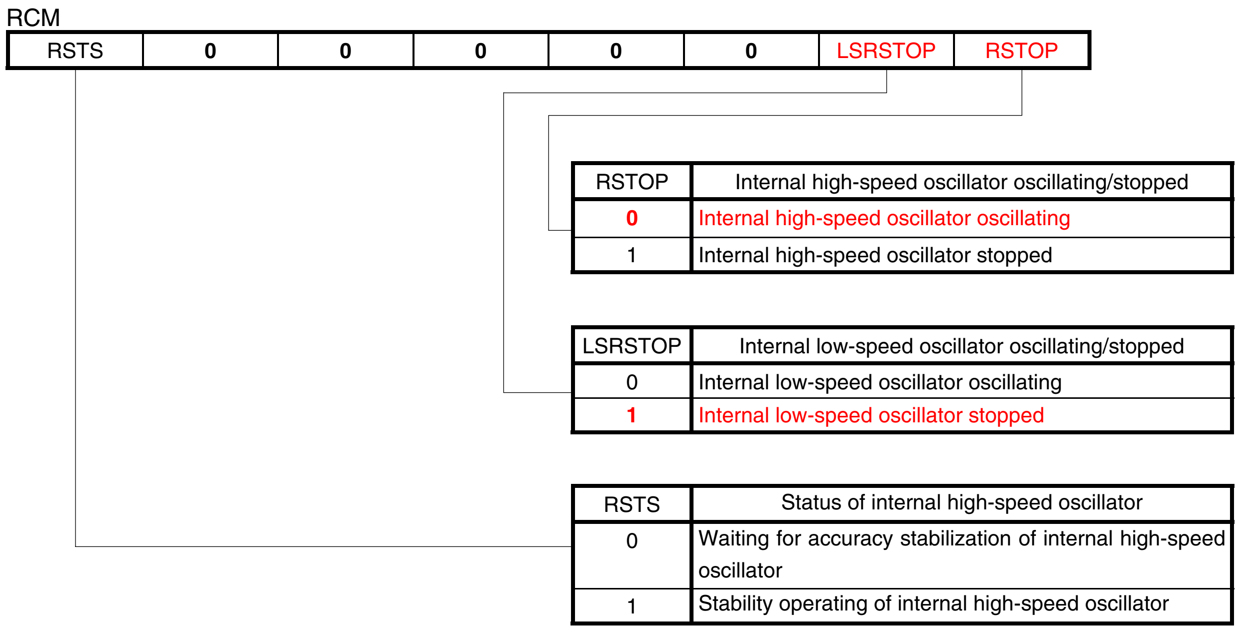
Remarks 1. x: don't care

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

(4) Specifying the operating mode of the internal oscillators

Specify the operating mode of the internal high-speed and low-speed oscillators by using the internal oscillation mode register (RCM).

Figure 4-8. Format of Internal Oscillation Mode Register (RCM)



RCM

RSTS	**0**	**0**	**0**	**0**	**0**	LSRSTOP	RSTOP

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

RSTS	Status of internal high-speed oscillator
0	Waiting for accuracy stabilization of internal high-speed oscillator
1	Stability operating of internal high-speed oscillator

Cautions 1. The value of this register is 00H immediately after a reset ends, but automatically changes to 80H after internal high-speed oscillator has been stabilized.

2. Bit 7 is read-only.

3. Be sure to clear bits 6 to 2 to "0".

4. When setting RSTOP to 1, be sure to confirm that the CPU runs on a clock other than the internal high-speed oscillation clock. Specifically, set under either of the following conditions.

a. 78K0/KY2-L, 78K0/KA2-L, and 78K0/KB2-L

- **When MCS = 1 (when the CPU runs on the high-speed system clock)**

b. 78K0/KC2-L

- **When MCS = 1 and CLS = 0 (when the CPU runs on the high-speed system clock)**
- **When CLS = 1 (when the CPU runs on the subsystem clock)**

In addition, stop peripheral hardware that runs on the internal high-speed oscillation clock before setting RSTOP to 1.

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

(5) Specifying the operating mode of the high-speed system clock

Specify the operating mode of the high-speed system clock by using the main OSC control register (MOC).

Figure 4-9. Format of Main OSC Control Register (MOC)

MOC

MSTOP	**0**	**0**	**0**	**0**	**0**	**0**	**0**

MSTOP	Control of high-speed system clock operation	
	X1 oscillation mode	External clock input mode
0	X1 oscillator operating	External clock from EXCLK pin is enabled
1	X1 oscillator stopped	External clock from EXCLK pin is disabled

Cautions 1. When setting MSTOP to 1, be sure to confirm that the CPU runs on a clock other than the high-speed system clock. Specifically, set under either of the following conditions.

a. 78K0/KY2-L, 78K0/KA2-L, and 78K0/KB2-L

- **When MCS = 0 (when the CPU runs on the internal high-speed oscillation clock)**

b. 78K0/KC2-L

- **When MCS = 0 and CLS = 0 (when the CPU runs on the internal high-speed oscillation clock)**
- **When CLS = 1 (when the CPU runs on the subsystem clock)**

In addition, stop peripheral hardware that runs on the high-speed system clock before setting MSTOP to 1.

2. Do not clear MSTOP to 0 while bit 6 (OSCSEL) of the clock operation mode select register (OSCCTL) is 0 (input port mode).

3. The peripheral hardware cannot operate when the peripheral hardware clock is stopped. To resume the operation of the peripheral hardware after the peripheral hardware clock has been stopped, initialize the peripheral hardware.

4. Be sure to clear bits 6 to 0 to "0".

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

(6) Selecting the main system clock and the clock to supply to the peripheral hardware

Select the main system clock to supply to the CPU and the clock to supply to the peripheral hardware by using the main clock mode register (MCM).

Figure 4-10. Format of Main Clock Mode Register (MCM)

MCM

0	0	0	0	0	XSEL	MCS	MCM0

XSEL	MCM0	Selection of clock supplied to main system clock and peripheral hardware	
		Main system clock (f_{XP})	Peripheral hardware clock (f_{PRS})
0	**0**	Internal high-speed oscillation clock (f_{RH})	Internal high-speed oscillation clock (f_{RH})
0	1		
1	0		High-speed system clock (f_{XH})
1	1	High-speed system clock (f_{XH})	

MCS	Main system clock status
0	Runs on the internal high-speed oscillation clock.
1	Runs on the high-speed system clock.

Cautions 1. Bit 1 is read-only.

2. XSEL can be changed only once immediately after a reset ends.

3. Do not rewrite MCM0 when the CPU runs on the subsystem clock.

4. A clock other than f_{PRS} is supplied to the following peripherals regardless of the settings of XSEL and MCM0.

- **Watchdog timer (that runs on the internal low-speed oscillation clock)**
- **When "f_{IL}", "$f_{IL}/2^6$", or "$f_{IL}/2^{15}$" is selected as the count clock for 8-bit timer H1 (that runs on the internal low-speed oscillation clock)**
- **Peripheral hardware for which an external clock is selected as the clock source (except when the external count clock of TM00 is selected (TI000 pin valid edge))**

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

(7) Controlling the real-time counter control clock

Control the real-time counter[Note 2] control clock by using peripheral enable register 0 (PER0)[Note 1].

Figure 4-11. Format of Peripheral Enable Register 0 (PER0)

PER0

RTCEN	0	0	0	0	0	0	0

RTCEN	Control of real-time counter (RTC) control clock
0	Stops supply of control clock.[Note 2] • SFR used by the real-time counter cannot be written (can be read). • Operation of the real-time counter continues.
1	Supplies control clock. • SFR used by the real-time counter can be read and written.

Notes 1. The PER0 register is used only in the 78K0/KC2-L.

2. Power consumption can be further reduced by stopping the supply of the real-time counter control clock.

Cautions 1. The control clock supply stopped by clearing RTCEN to 0 is the clock to be used when write-accessing the registers (such as the RTCC0 register) to be used for the real-time counter (RTC) from the CPU. The RTC operating clock (f_{SUB}) does not stop, even if RTCEN is cleared to 0.

2. Be sure to clear bits 6 to 0 to "0".

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

An example of writing the values specified in (1) to (7) in the program is shown below.

• Assembly language

```
MOV    OSCCTL, #00000000B
MOV    PCC,    #00000001B
MOV    RCM,    #00000010B
MOV    MOC,    #10000000B
MOV    MCM,    #00000000B
MOV    PER0,   #00000000B
```

• C language

```
OSCCTL = 0b00000000;
PCC    = 0b00000001;
RCM    = 0b00000010;
MOC    = 0b10000000;
MCM    = 0b00000000;
PER0   = 0b00000000;
```

4.7 Setting Up Ports

Caution The on-chip ports vary depending on the product, so the ports to set up also vary.

	78K0/KY2-L	78K0/KA2-L	78K0/KB2-L	78K0/KC2-L	
				44-Pin Product	48-Pin Product
Port 0	P00, P01	P00, P01	P00, P01	P00, P01	P00 to P02
Port 1	–	–	P10 to P17	P10 to P17	P10 to P17
Port 2	P20 to P23	P20 to P25	P20 to P23	P20 to P27	P20 to P27
Port 3	P30	P30 to P32	P30 to P33	P30 to P33	P30 to P33
Port 4	–	–	–	P40, P41	P40 to P42
Port 6	P60, P61	P60, P61	P60, P61	P60 to P63	P60 to P63
Port 7	–	–	–	P70 to P73	P70 to P75
Port 12	P121 to P122, P125	P121 to P122, P125	P120 to P122, P125	P120 to P125	P120 to P125

(1) Specifying ports as input or output ports

The PMxx registers are used to specify whether ports are used as input ports or output ports. Ports are specified as input ports immediately after a reset ends.

The PMxx format is described, taking the PM0 register as an example.

Figure 4-12. Format of Port Mode Register 0 (PM0)

PM0

1	1	1	1	1	PM02[Note]	PM01	PM00

PM0n (n = 0 to 2)	Selection of P0n (n = 0 to 2) pin I/O mode
0	Output mode (output buffer on)
1	Input mode (output buffer off)

Note This bit is used only in 78K0/KC2-L 48-pin products.

Caution Be sure to set the following bits to 1:

- **78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L, 78K0/KC2-L (44-pin products): Bits 7 to 2**
- **78K0/KC2-L (48-pin products): Bits 7 to 3**

(2) Setting up the output latches of output ports

The Pxx registers are used to set up the output latches of output ports to high level or low level. The output latches of output ports are set to low-level output immediately after a reset ends.

The Pxx format is described, taking the P0 register as an example.

Figure 4-13. Format of Port Register 0 (P0)

P0

0	0	0	0	0	P02[Note]	P01	P00

P0n (n = 0 to 2)	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 This bit is used only in 78K0/KC2-L 48-pin products.

Caution Be sure to clear the following bits to 0:

- **78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L, 78K0/KC2-L (44-pin products): Bits 7 to 2**
- **78K0/KC2-L (48-pin products): Bits 7 to 3**

(3) Specifying the connections of internal pull-up resistors to input ports

The PUxx registers are used to specify whether internal pull-up resistors are connected to input ports. Internal pull-up resistors are not connected immediately after a reset ends.

The PUxx format is described, taking the PU0 register as an example.

Figure 4-14. Format of Pull-up Resistor Option Register 0 (PU0)

PU0

0	0	0	0	0	PU02[Note]	PU01	PU00

PU0n (n = 0 to 2)	Connection of P0n (n = 0 to 2) pin internal pull-up resistor
0	Internal pull-up resistor is not connected.
1	Internal pull-up resistor is connected.

Note This bit is used only in 78K0/KC2-L 48-pin products.

Caution Be sure to clear the following bits to 0:

- **78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L, 78K0/KC2-L (44-pin products): Bits 7 to 2**
- **78K0/KC2-L (48-pin products): Bits 7 to 3**

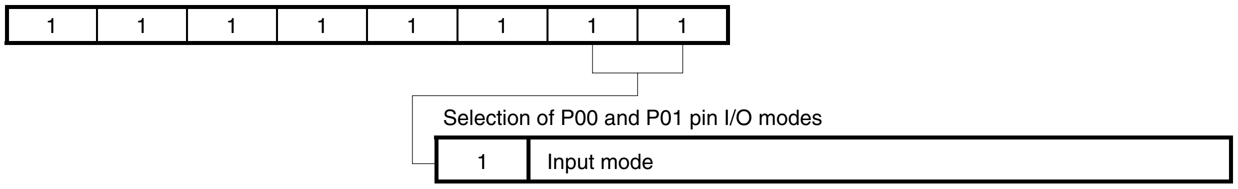
[Example 1] P0 is set up as follows to use it for switch input:

- P00 and P01 are specified as input ports.
- An internal pull-up resistor is connected to P00 and P01.

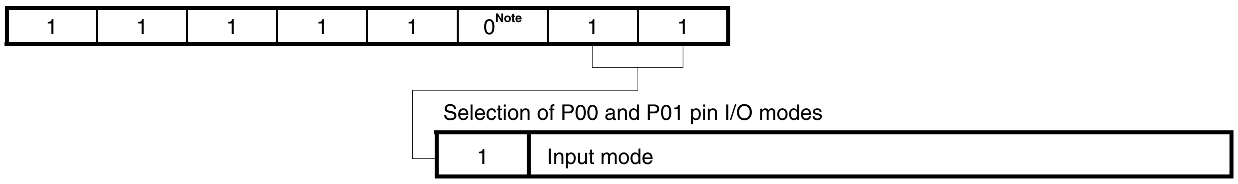
(Same as in the sample program settings)

PM0

- 78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L, 78K0/KC2-L (44-pin products)

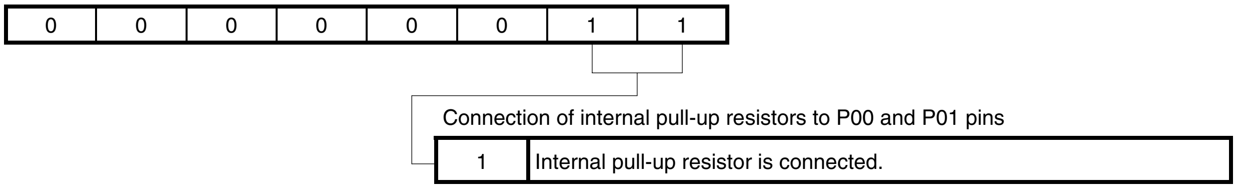


- 78K0/KC2-L (48-pin products)



Note Unused pins are assumed to be output port pins.

PU0


0 0 0 0 0 0 1 1
Connection of internal pull-up resistors to P00 and P01 pins
1 Internal pull-up resistor is connected.

In this sample program, switch input signals (SW1 and SW2) are used as low-active signals. Therefore, a low-level signal (0) is input to the switch input ports (P00 and P01) when the switches are turned on and a high-level signal (1) is input when the switches are turned off.

The relationship between the switch input signals (SW1 and SW2) and switch input ports (P00 and P01) is as follows:

Switch Input		Switch Input Port	
SW1	SW2	P00	P01
On	On	0	0
Off	On	1	0
On	Off	0	1
Off	Off	1	1

These settings are specified in the program as follows:

• 78K0/KY2-L, 78K0/KA2-L, 78K0/KB2-L, 78K0/KC2-L (44-pin products)

[Assembly language]

```
MOV    P0,    #00000000B
MOV    PM0,   #11111111B
MOV    PU0,   #00000011B
```

[C language]

```
P0     = 0b00000000;
PM0    = 0b11111111;
PU0    = 0b00000011;
```

• 78K0/KC2-L (48-pin products)

[Assembly language]

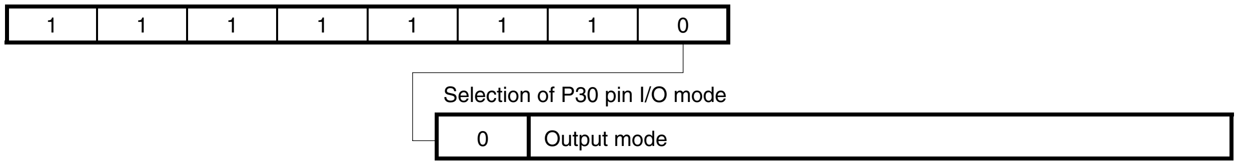
```
MOV    P0,    #00000000B
MOV    PM0,   #11111011B
MOV    PU0,   #00000011B
```

[C language]

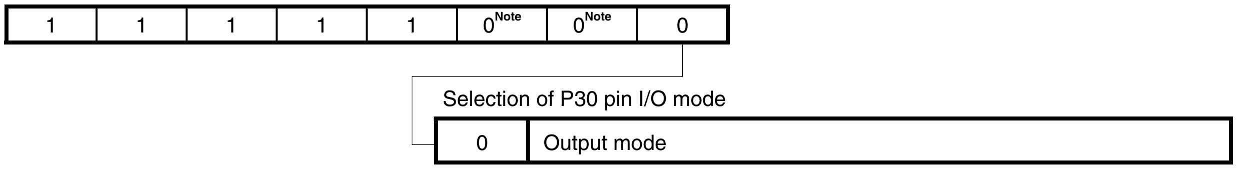
```
P0     = 0b00000000;
PM0    = 0b11111011;
PU0    = 0b00000011;
```

[Example 2] P3 and P6 are set up as follows to turn on the LEDs:
- P30, P60, and P61 are specified as output ports.
- The P30, P60, and P61 output latches are set to high-level output.
  (Same as in the sample program settings)

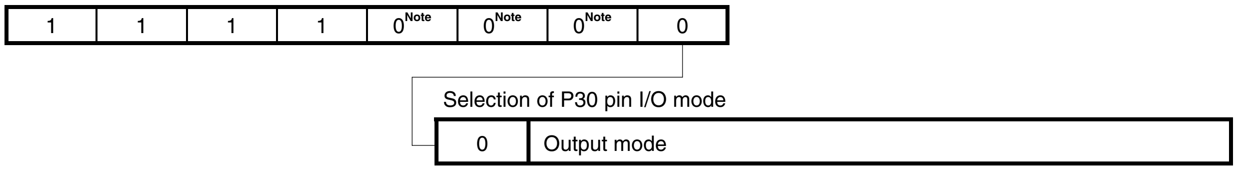
PM3
- 78K0/KY2-L



- 78K0/KA2-L

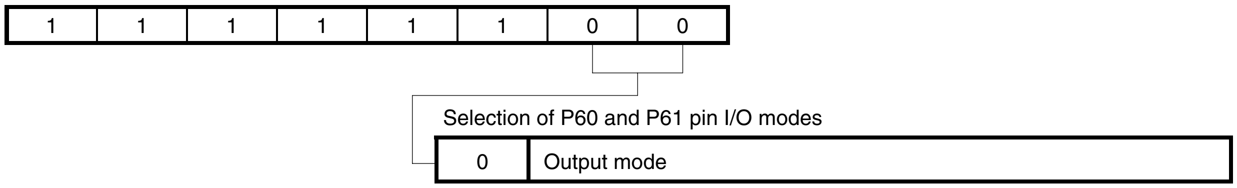


- 78K0/KB2-L and 78K0/KC2-L

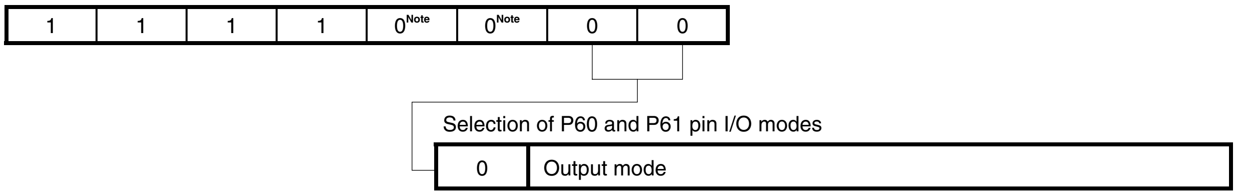


Note Unused pins are assumed to be output port pins.

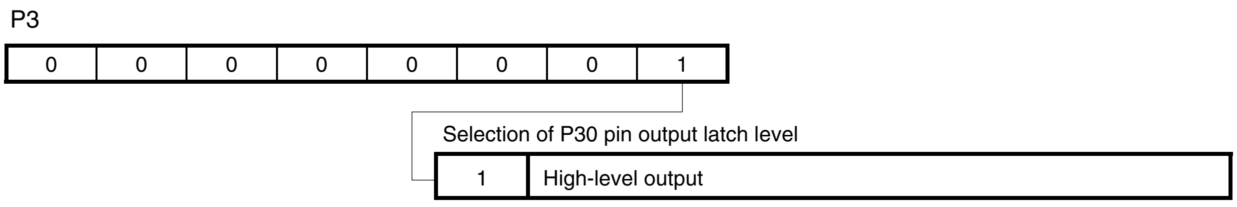
PM6
- 78K0/KY2-L, 78K0/KA2-L, and 78K0/KB2-L

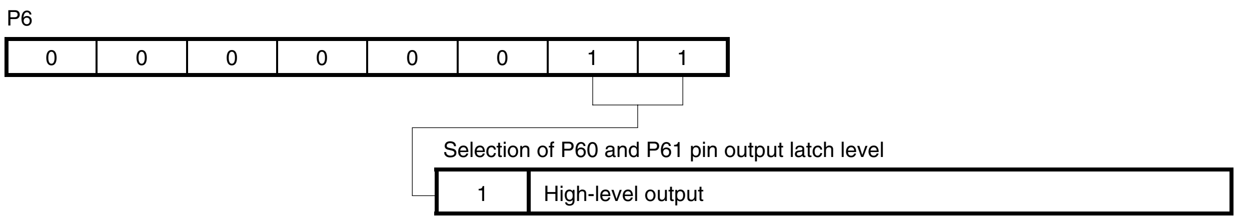


- 78K0/KC2-L



Note Unused pins are assumed to be output port pins.





In this sample program, the signals used to turn on the LEDs (LED1, LED2, and LED3) are used as active-low signals. Therefore, the LEDs are turned on if 0 is output from the LED output ports (P30, P60, and P61) or turned off if 1 is output from the ports.

The relationship between the values of the LED output ports (P30, P60, and P61) and whether the LEDs are turned on (LED1, LED2, and LED3) is as follows:

LED Output Port			LED Status		
P30	P60	P61	LED1	LED2	LED3
0	1	1	On	Off	Off
1	0	1	Off	On	Off
1	1	0	Off	Off	On
1	1	1	Off	Off	Off

These settings are specified in the program as follows:

• 78K0/KY2-L
[Assembly language]

```
MOV    P3,    #00000001B
MOV    PM3,   #11111110B
MOV    P6,    #00000011B
MOV    PM6,   #11111100B
```

[C language]

```
P3     = 0b00000001;
PM3    = 0b11111110;
P6     = 0b00000011;
PM6    = 0b11111100;
```

• 78K0/KA2-L
[Assembly language]

```
MOV    P3,    #00000001B
MOV    PM3,   #11111000B
MOV    P6,    #00000011B
MOV    PM6,   #11111100B
```

[C language]

```
P3     = 0b00000001;
PM3    = 0b11111000;
P6     = 0b00000011;
PM6    = 0b11111100;
```

• 78K0/KB2-L
[Assembly language]

```
MOV    P3,    #00000001B
MOV    PM3,   #11110000B
MOV    P6,    #00000011B
MOV    PM6,   #11111100B
```

[C language]

```
P3     = 0b00000001;
PM3    = 0b11110000;
P6     = 0b00000011;
PM6    = 0b11111100;
```

• 78K0/KC2-L
[Assembly language]

```
MOV    P3,    #00000001B
MOV    PM3,   #11110000B
MOV    P6,    #00000011B
MOV    PM6,   #11110000B
```

[C language]

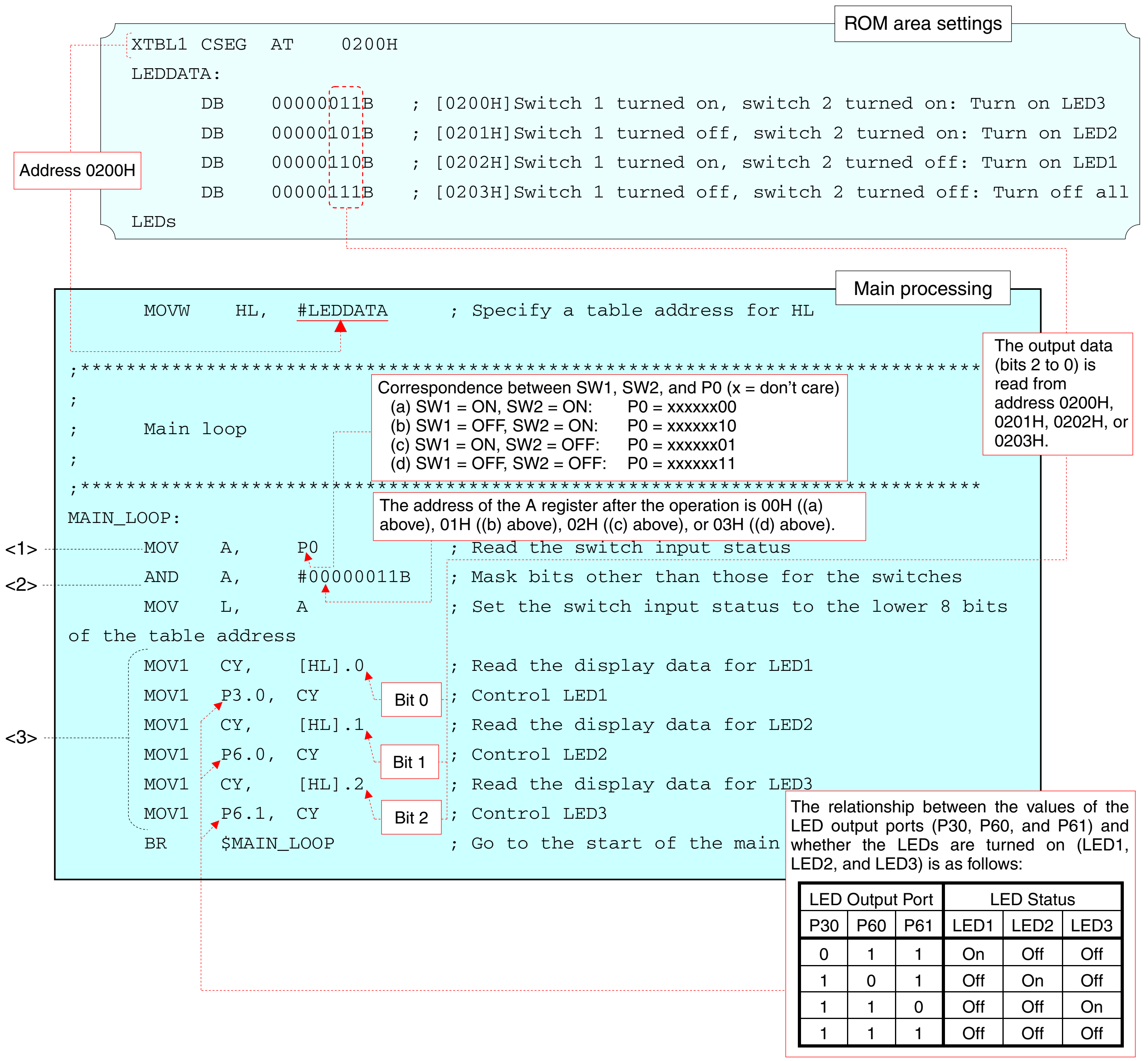
```
P3     = 0b00000001;
PM3    = 0b11110000;
P6     = 0b00000011;
PM6    = 0b11110000;
```

4.8 Main Processing

The following operations are performed with the main processing in assembly language.

<1> Read data from P0.
<2> Among the read eight bits, clear the bits other than those for the switch input ports (P00 and P01) to 0.
<3> Read the data to output in accordance with the combination of the P00 and P01 input levels from addresses 0200H to 0203H (in the LEDDATA table), and then sequentially output the values of each bit to P30, P60, and P61.

By performing operations <1> and <2>, only the combination of the inputs of the switches (SW1 and SW2) connected to P00 and P01 can be determined. In this sample program, the signals from the switches are used as active-low signals. Therefore, a low-level signal (0) is input to P00 and P01 if the switches are turned on and a high-level signal (1) is input to P00 and P01 if the switches are turned off.



ROM area settings

```
XTBL1 CSEG   AT     0200H
LEDDATA:
        DB      00000011B    ; [0200H]Switch 1 turned on, switch 2 turned on: Turn on LED3
        DB      00000101B    ; [0201H]Switch 1 turned off, switch 2 turned on: Turn on LED2
        DB      00000110B    ; [0202H]Switch 1 turned on, switch 2 turned off: Turn on LED1
        DB      00000111B    ; [0203H]Switch 1 turned off, switch 2 turned off: Turn off all
LEDs
```

Address 0200H

Main processing

```
        MOVW    HL,     #LEDDATA        ; Specify a table address for HL

;**********************************************************************
;
;       Main loop
;
;**********************************************************************
MAIN_LOOP:
        MOV     A,      P0              ; Read the switch input status
        AND     A,      #00000011B      ; Mask bits other than those for the switches
        MOV     L,      A               ; Set the switch input status to the lower 8 bits
of the table address
        MOV1    CY,     [HL].0          ; Read the display data for LED1
        MOV1    P3.0,   CY              ; Control LED1
        MOV1    CY,     [HL].1          ; Read the display data for LED2
        MOV1    P6.0,   CY              ; Control LED2
        MOV1    CY,     [HL].2          ; Read the display data for LED3
        MOV1    P6.1,   CY              ; Control LED3
        BR      $MAIN_LOOP              ; Go to the start of the main
```

Correspondence between SW1, SW2, and P0 (x = don't care)

(a) SW1 = ON, SW2 = ON:	P0 = xxxxxx00
(b) SW1 = OFF, SW2 = ON:	P0 = xxxxxx10
(c) SW1 = ON, SW2 = OFF:	P0 = xxxxxx01
(d) SW1 = OFF, SW2 = OFF:	P0 = xxxxxx11

The address of the A register after the operation is 00H ((a) above), 01H ((b) above), 02H ((c) above), or 03H ((d) above).

The output data (bits 2 to 0) is read from address 0200H, 0201H, 0202H, or 0203H.

The relationship between the values of the LED output ports (P30, P60, and P61) and whether the LEDs are turned on (LED1, LED2, and LED3) is as follows:

LED Output Port			LED Status		
P30	P60	P61	LED1	LED2	LED3
0	1	1	On	Off	Off
1	0	1	Off	On	Off
1	1	0	Off	Off	On
1	1	1	Off	Off	Off

The main processing in C language operates similarly to that in assembly language.
In C language, the correspondence between the input data and output data is specified as an array.

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

     Main loop

******************************************************************************/
void main(void)
{
    const unsigned char aLedOut[4]
     = {0b00000011,0b00000101,0b00000110,0b00000111};    /* Table for turning on the LEDs */
    unsigned char ucSwitchBuffer;                         /* Switch input data storage area */

    while(1){
       /* Acquire valid switch information */
       ucSwitchBuffer = ( P0 & 0b00000011 );

       /* Read the data to display from the table and display */
       P3 = ( aLedOut[ucSwitchBuffer] & 0b00000001 );         /* Turn on LED1 */
       P6 = ( ( aLedOut[ucSwitchBuffer] >> 1 ) & 0b00000011 ); /* Turn on LED2 and LED3 */
    }
}
```

Four units of data are defined in the brackets wherein output data is specified.

The correspondences between the input data and output data are as follows:

Switch Input	P00, P01	ucSwitchBuffer	aLedOut	LED Status
SW1 = ON, SW2 = ON	P00 = 0, P01 = 0	0b00000000	0b00000011	Turn on only LED3.
SW1 = OFF, SW2 = ON	P00 = 1, P01 = 0	0b00000001	0b00000101	Turn on only LED2.
SW1 = ON, SW2 = OFF	P00 = 0, P01 = 1	0b00000010	0b00000110	Turn on only LED1.
SW1 = OFF, SW2 = OFF	P00 = 1, P01 = 1	0b00000011	0b00000111	Turn off all LEDs.

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

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 (Initial Settings)
;******************************************************************************
;    LED Lighting Switch Control
;******************************************************************************
;<<History>>
;    2009.1.--    Release
;******************************************************************************
;
;<<Overview>>
; This sample program initializes the microcontroller by specifying settings such as
; selecting the clock frequency and setting up I/O ports.  After the initialization,
; three LED lights are controlled by two switches in the main loop.
;
; <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
; - Specifying that the CPU clock run on the internal high-speed oscillation clock (4 MHz)
; - Stopping the internal low-speed oscillator
; - Setting up I/O ports
; - Disabling peripheral hardware not to be used
;
;
; <Switch input and LED status>
;
;  +--------------------------------------------------+
;  | Switch 1| Switch 2|   LED1   |   LED2   |   LED3   |
;  |  (P00)  |  (P01)  |  (P30)   |  (P60)   |  (P61)   |
;  |-------------------|------------------------------|
;  |   OFF   |   OFF   |   OFF    |   OFF    |   OFF    |
;  |   ON    |   OFF   |   ON     |   OFF    |   OFF    |
;  |   OFF   |   ON    |   OFF    |   ON     |   OFF    |
;  |   ON    |   ON    |   OFF    |   OFF    |   ON     |
  ;  +-------------------------------------------------+
  ;  * 0 is input to the ports if the switches are turned on and 1 is input to the ports
if the switches are turned off.
  ;  * The LEDs are turned off if 1 is output from the ports or turned on if 0 is output
from the ports.
  ;
  ;
  ;<<I/O port settings>>
  ;  Input: P00, P01
  ;  Output: P30, 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
;
;===============================================================================
XTBL1CSEG   AT     0200H
LEDDATA:
      DB     00000011B          ; [0200H]Switch 1 turned on, switch 2 turned on: Turn on
LED3
      DB     00000101B          ; [0201H]Switch 1 turned off, switch 2 turned on: Turn
on LED2
      DB     00000110B          ; [0202H]Switch 1 turned on, switch 2 turned off: Turn
on LED1
      DB     00000111B          ; [0203H]Switch 1 turned off, switch 2 turned off: Turn
off all LEDs

;===============================================================================
;
;     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
;
;*******************************************************************************
XMAINCSEG   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

;-----------------------------------------------------------------------------
;    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,   #00000001B   ; Select the CPU clock (fCPU)
 ;                  |||+|+++------- CSS/PCC2/PCC1/PCC0
 ;                  ||| |          [CPU clock (fCPU) selection]
 ;                  ||| |           0000:fXP
 ;                  ||| |           0001:fXP/2
 ;                  ||| |           0010:fXP/2^2
 ;                  ||| |           0011:fXP/2^3
 ;                  ||| |           0100:fXP/2^4
 ;                  ||| |           1000:fSUB/2
 ;                  ||| |           1001:fSUB/2
 ;                  ||| |           1010:fSUB/2
 ;                  ||| |           1011:fSUB/2
 ;                  ||| |           1100:fSUB/2
 ;                  ||| |           (Other than the above: Setting prohibited)
 ;                  ||| +---------- Be sure to clear this bit to 0
 ;                  ||+------------ CLS
 ;                  ||             [CPU clock status]
 ;                  |+------------- XTSTART
 ;                  |              [Subsystem clock pin operation setting]
 ;                  |               Specify the use of the pin by also using EXCLKS and OSCSELS
 ;                  +-------------- Be sure to clear this bit to 0

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

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

     MOV    MCM,   #00000000B    ; Select the clock to supply
;                   |||||+|+------- XSEL/MCM0:
;                   ||||| |        [Clock supplied to main system and
;                   ||||| |         peripheral hardware]
;                   ||||| |         00: Main system clock (fXP)
;                   ||||| |              = internal high-speed oscillation clock (fIH)
;                   ||||| |             Peripheral hardware clock (fPRS)
;                   ||||| |              = internal high-speed oscillation clock (fIH)
;                   ||||| |         01: Main system clock (fXP)
;                   ||||| |              = internal high-speed oscillation clock (fIH)
;                   ||||| |             Peripheral hardware clock (fPRS)
;                   ||||| |              = internal high-speed oscillation clock (fIH)
;                   ||||| |         10: Main system clock (fXP)
;                   ||||| |              = internal high-speed oscillation clock (fIH)
;                   ||||| |             Peripheral hardware clock (fPRS)
;                   ||||| |              = high-speed system clock (fIH)
;                   ||||| |         11: Main system clock (fXP)
;                   ||||| |              = high-speed system clock (fIH)
;                   ||||| |             Peripheral hardware clock (fPRS)
;                   ||||| |              = high-speed system clock (fIH)
;                   ||||| +-------- MCS
;                   |||||          [Main system clock status]
;                   +++++---------- Be sure to clear this bit to 0

     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

;-------------------------------------------------------------------------------
;    Initialize port 0
;-------------------------------------------------------------------------------
     MOV    P0,    #00000000B    ; Set the P00 to P02 output latches to low level
     MOV    PM0,   #11111011B    ; Specify P00 and P01 as input ports
                                 ; Specify P02 as an output port
     MOV    PU0,   #00000011B    ; Connect internal pull-up resistors to P00 and P01
                                 ; Does not connect an internal pull-up resistor to P02
```

```
                                ; P00: Use for switch 1 input
                                ; P01: Use for switch 2 input
                                ; 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,    #00000001B   ; Set the P30 output latch to high level
                                ; Set the P31 to P33 output latches to low level
     MOV    PM3,   #11110000B   ; Specify P30 to P33 as output ports
                                ; P30: Use for turning on LED1
                                ; P31 to P33: Unused

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

;--------------------------------------------------------------------------------
;    Initialize port 6
;--------------------------------------------------------------------------------
     MOV    P6,    #00000011B   ; Set the P60 and P61 output latches to high level
                                ; Set the P62 and P63 output latches to low level
     MOV    PM6,   #11110000B   ; Specify P60 to P63 as output ports
                                ; P60: Use for turning on LED2
                                ; P61: Use for turning on LED3
                                ; 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

;-------------------------------------------------------------------------------
;    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 timers H0 and H1
     MOV    TMHMD0,      #00000000B   ; Stop timer H0
     MOV    TMHMD1,      #00000000B   ; Stop timer H1

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

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

```
      ; Low-voltage detector
      MOV    LVIM,  #00000000B   ; Disable the detector

      ; 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

 ;-------------------------------------------------------------------------------
 ;    Initialize the general-purpose register
 ;-------------------------------------------------------------------------------
      MOVW   HL,    #LEDDATA     ; Specify the table address for turning on the LEDs

 ;-------------------------------------------------------------------------------
 ;    Enable interrupts
 ;    (To use interrupts, enable interrupts here.)
 ;-------------------------------------------------------------------------------
 ;    EI                         ; To enable interrupts,
                                 ; uncomment this line.

      BR     MAIN_LOOP           ; Go to the main loop


 ;*******************************************************************************
 ;
 ;    Main loop
 ;
 ;*******************************************************************************
 MAIN_LOOP:
      MOV    A,     P0           ; Read the switch input status
      AND    A,     #00000011B   ; Mask bits other than those for the switches
      MOV    L,     A            ; Set the switch input status to the lower 8 bits of the
table address
      MOV1   CY,    [HL].0       ; Read the display data for LED1
      MOV1   P3.0,  CY           ; Control LED1
      MOV1   CY,    [HL].1       ; Read the display data for LED2
      MOV1   P6.0,  CY           ; Control LED2
      MOV1   CY,    [HL].2       ; Read the display data for LED3
      MOV1   P6.1,  CY           ; Control LED3
      BR     $MAIN_LOOP          ; Go to the start of the main loop
 end
```

● main.c (C language version)

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

  NEC Electronics    78K0/KC2-L Series

*******************************************************************************
  78K0/KC2-L Series   Sample Program (Initial Settings)
*******************************************************************************
  LED Lighting Switch Control
*******************************************************************************
<<History>>
  2009.1.-- Release
*******************************************************************************

<<Overview>>
This sample program initializes the microcontroller by specifying settings such as
selecting the clock frequency and setting up I/O ports.  After the initialization,
three LED lights are controlled by two switches in the main loop.

 <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
 - Specifying that the CPU clock run on the internal high-speed oscillation clock (4 MHz)
 - Stopping the internal low-speed oscillator
 - Setting up I/O ports
 - Disabling peripheral hardware not to be used


 <Switch input and LED status>

  +-------------------------------------------------+
  | Switch 1| Switch 2|   LED1   |   LED2   |   LED3   |
  |  (P00)   |  (P01)   |  (P30)   |  (P60)   |  (P61)   |
  |-------------------|-----------------------------|
  |   OFF    |   OFF    |   OFF    |   OFF    |   OFF    |
  |   ON     |   OFF    |   ON     |   OFF    |   OFF    |
  |   OFF    |   ON     |   OFF    |   ON     |   OFF    |
  |   ON     |   ON     |   OFF    |   OFF    |   ON     |
  +-------------------------------------------------+
  * 0 is input to the ports if the switches are turned on and 1 is input to the ports if
the switches are turned off.
  * The LEDs are turned off if 1 is output from the ports or turned on if 0 is output from
the ports.
```

```
<<I/O port settings>>
  Input: P00, P01
  Output: P30, 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 */

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

  Initialization after RESET

******************************************************************************/
void hdwinit( void )
{
/*----------------------------------------------------------------------------
  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 */

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

```
   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    = 0b00000001; /* Select the CPU clock (fCPU) */
   /*          |||+|+++---- CSS/PCC2/PCC1/PCC0 */
   /*          ||| |      [CPU clock (fCPU) selection] */
   /*          ||| |       0000:fXP */
   /*          ||| |       0001:fXP/2 */
   /*          ||| |       0010:fXP/2^2 */
   /*          ||| |       0011:fXP/2^3 */
   /*          ||| |       0100:fXP/2^4 */
   /*          ||| |       1000:fSUB/2 */
   /*          ||| |       1001:fSUB/2 */
   /*          ||| |       1010:fSUB/2 */
   /*          ||| |       1011:fSUB/2 */
   /*          ||| |       1100:fSUB/2 */
   /*          ||| |       (Other than the above: Setting prohibited) */
   /*          ||| +------- Be sure to clear this bit to 0 */
   /*          ||+--------- CLS */
   /*          ||         [CPU clock status] */
   /*          |+---------- XTSTART */
   /*          |          [Subsystem clock pin operation setting] */
   /*          |           Specify the use of the pin by also using EXCLKS and OSCSELS */
   /*          +----------- Be sure to clear this bit to 0 */

   RCM    = 0b00000010; /* Select the operating mode of the internal oscillator */
```

```
    /*          |||||||+---- RSTOP */
    /*          |||||||    [Internal high-speed oscillator oscillating/stopped] */
    /*          |||||||     0: Internal high-speed oscillator oscillating */
    /*          |||||||     1: Internal high-speed oscillator stopped */
    /*          ||||||+----- LSRSTOP */
    /*          ||||||     [Internal low-speed oscillator oscillating/stopped] */
    /*          ||||||      0: Internal low-speed oscillator oscillating */
    /*          ||||||      1: Internal low-speed oscillator stopped */
    /*          |+++++------ Be sure to clear this bit to 0 */
    /*          +----------- RSTS */
    /*                     [Status of internal high-speed oscillator] */

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

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

    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 */

/*-----------------------------------------------------------------------------
  Initialize port 0
-----------------------------------------------------------------------------*/
  P0     = 0b00000000;/* Set the P00 to P02 output latches to low level */
  PM0    = 0b11111011;/* Specify P00 and P01 as input ports */
                      /* Specify P02 as an output port */
  PU0    = 0b00000011;/* Connect internal pull-up resistors to P00 and P01 */
                      /* Does not connect an internal pull-up resistor to P02 */
                      /* P00: Use for switch 1 input */
                      /* P01: Use for switch 2 input */
                      /* 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     = 0b00000001;/* Set the P30 output latch to high level */
                      /* Set the P31 to P33 output latches to low level */
  PM3    = 0b11110000;/* Specify P30 to P33 as output ports */
                      /* P30: Use for turning on LED1 */
                      /* P31 to P33: Unused */

/*-----------------------------------------------------------------------------
  Initialize port 4
-----------------------------------------------------------------------------*/
  P4     = 0b00000000;/* Set the P40 to P42 output latches to low level */
  PM4    = 0b11111000;/* Specify P40 to P42 as output ports */
                      /* P40 to P42: Unused */

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

```
  Initialize port 6
-------------------------------------------------------------------------------*/
  P6     = 0b00000011; /* Set the P60 and P61 output latches to high level */
                       /* Set the P62 and P63 output latches to low level */
  PM6    = 0b11110000; /* Specify P60 to P63 as output ports */
                       /* P60: Use for turning on LED2 */
                       /* P61: Use for turning on LED3 */
                       /* 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 */

/*-------------------------------------------------------------------------------
  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 timers H0 and H1 */
  TMHMD0 = 0b00000000; /* Stop timer H0 */
  TMHMD1 = 0b00000000; /* Stop timer H1 */

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

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

  /* Low-voltage detector */
  LVIM   = 0b00000000; /* Disable the detector */

  /* 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 */

/*-------------------------------------------------------------------------------
  Enable interrupts
  (To use interrupts, enable interrupts here.)
-------------------------------------------------------------------------------*/
/*   EI();  */            /* To enable interrupts,   */
                      /* uncomment this line. */

}


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

  Main loop

*******************************************************************************/
void main(void)
{
  const unsigned char aLedOut[4]
   = {0b00000011,0b00000101,0b00000110,0b00000111}; /* Table for turning on the LEDs */
  unsigned char ucSwitchBuffer;                  /* Switch input data storage area */

  while(1){
     /* Acquire valid switch information */
     ucSwitchBuffer = ( P0 & 0b00000011 );
```

```
        /* Read the data to display from the table and display */
        P3 = ( aLedOut[ucSwitchBuffer] & 0b00000001 );       /* Control LED1 */
        P6 = ( ( aLedOut[ucSwitchBuffer] >> 1 ) & 0b00000011 );  /* Control LED2 and LED3
*/
    }
  }
```

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

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

(1) Initial settings of ports

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

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

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

(2) Disabling unused peripheral hardware

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

APPENDIX C REVISION HISTORY

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

For further information,
please contact:

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

[America]

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

[Europe]

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

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

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

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

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

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

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

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

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

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

[Asia & Oceania]

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

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

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

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

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

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

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

G0706