To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Application Note

78K0S/Kx1+

Sample Program (8-bit Timer H1)

PWM Output

This document describes an operation overview of the sample program and how to use it, as well as how to set and use the PWM output function of 8-bit timer H1. In the sample program, the brightness of the LEDs is changed every 500 ms by using the PWM output function of 8-bit timer H1 to control the pulse output duty.

Target devices

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

CONTENTS

Document No. U18863EJ2V0AN00 (2nd edition)
Date Published September 2008 NS

CHAPTER 1 OVERVIEW

An example of using the PWM output function of 8-bit timer H1 is presented in this sample program. The brightness of the LEDs is changed every 500 ms by controlling the pulse output duty.

1.1 Main Contents of the Initial Settings

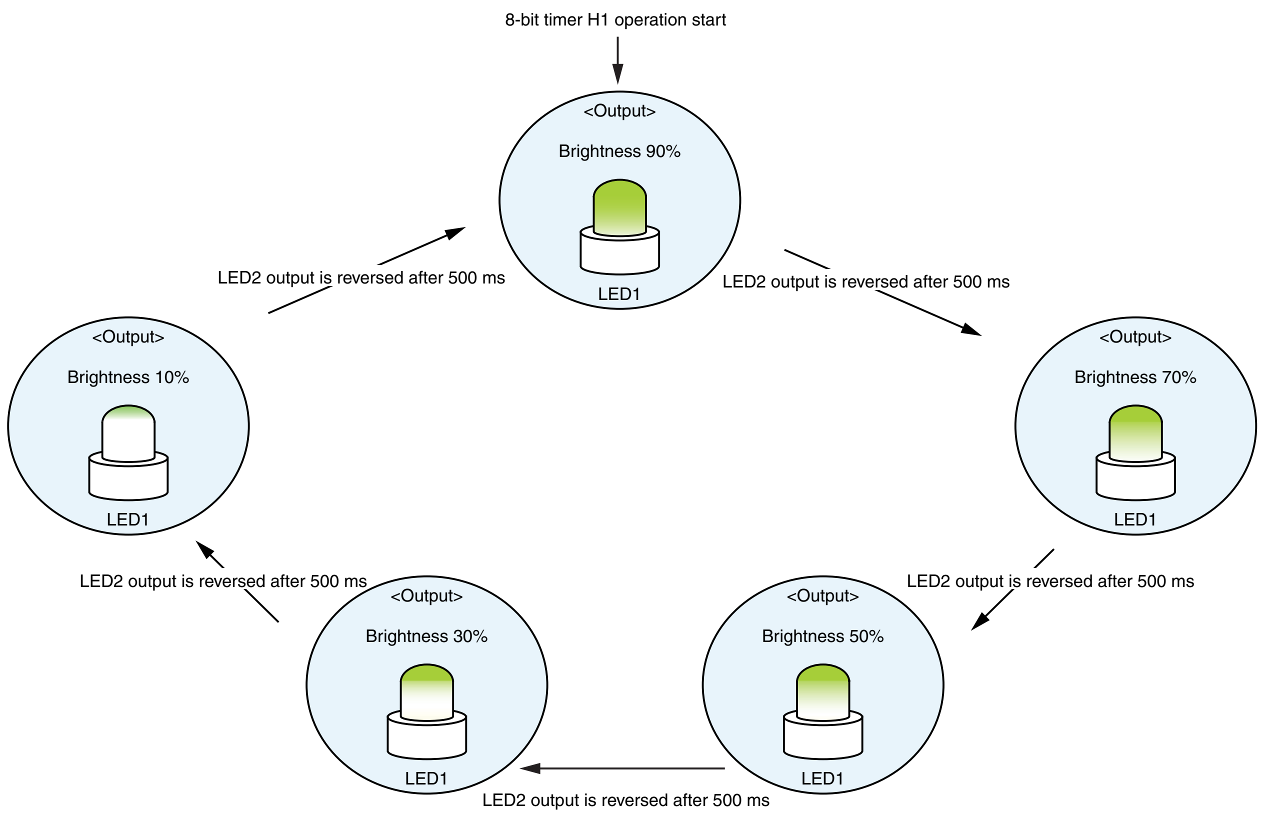
The main contents of the initial settings are as follows.

- Selecting the high-speed internal oscillator as the system clock source[Note]
- Stopping watchdog timer operation
- Setting V_{LVI} (low-voltage detection voltage) to 4.3 V ±0.2 V
- Generating an internal reset (LVI reset) signal when it is detected that V_{DD} is less than V_{LVI}, after V_{DD} (power supply voltage) becomes greater than or equal to V_{LVI}
- Setting the CPU clock frequency to 8 MHz
- Setting the I/O ports
- Setting 8-bit timer H1
  - Setting the count clock to $f_{XP}/2^6$ (125 kHz), setting the operation mode to the PWM output mode, enabling the timer output from TOH1, and setting the output level (default) to low level
  - Setting the PWM pulse output cycle to 2 ms (8 μs × 250) and the duty to 10%
- Enabling INTTMH1 interrupts

Note This is set by using the option byte.

1.2 Contents Following the Main Loop

The brightness of LED1 is changed by controlling the PWM output duty of 8-bit timer H1, after completion of the initial settings. The duty is changed every 500 ms by using the 8-bit timer H1 interrupts (INTTMH1). The LED2 output is reversed when the duty is changed.



PWM output duty	10%	30%	50%	70%	90%
LED1 brightness	90%	70%	50%	30%	10%

In this sample program, "LED1 brightness = 100 − duty" because the PWM output active level is set to high level and LED1 lights when it is at low level.

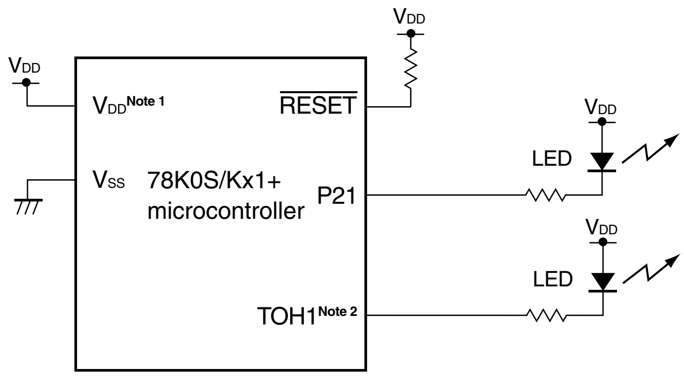
Caution For cautions when using the device, refer to the user's manual of each product (78K0S/KU1+, 78K0S/KY1+, 78K0S/KA1+, 78K0S/KB1+).

CHAPTER 2 CIRCUIT DIAGRAM

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

2.1 Circuit Diagram

A circuit diagram is shown below.



Notes 1. Use this in a voltage range of 4.5 V ≤ V_{DD} ≤ 5.5 V.

2. TOH1/P42: 78K0S/KA1+ and 78K0S/KB1+ microcontrollers

TOH1/ANI0/TI000/P20: 78K0S/KY1+ and 78K0S/KU1+ microcontrollers

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

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

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

2.2 Peripheral Hardware

The peripheral hardware to be used is shown below.

- LED1: PWM output
- LED2: Output reversal simultaneously with a PWM output duty change (reversed every 500 ms)

CHAPTER 3 SOFTWARE

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

3.1 File Configuration

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

File Name	Description	Compressed (*.zip) File Included		
				
main.asm (Assembly language version) main.c (C language version)	Source file for hardware initialization processing and main processing of microcontroller	● Note 1	● Note 1	
op.asm	Assembler source file for setting the option byte (sets the system clock source)	●	●	
tmh1pwm.prw	Work space file for integrated development environment PM+		●	
tmh1pwm.prj	Project file for integrated development environment PM+		●	
tmh1pwm.pri tmh1pwm.prs tmh1pwm.prm	Project files for system simulator SM+ for 78K0S/Kx1+		● Note 2	
tmh1pwm0.pnl	I/O panel file for system simulator SM+ for 78K0S/Kx1+ (used for checking peripheral hardware operations)		● Note 2	●
tmh1pwm0.wvo	Timing chart file for system simulator SM+ for 78K0S/Kx1+ (used for checking waveforms)			●

Notes 1. “main.asm” is included with the assembly language version, and “main.c” with the C language version.

2. These files are not included among the files for the 78K0S/KU1+ microcontroller.

Remark

 : Only the source file is included.

 : The files to be used with integrated development environment PM+ and 78K0S/Kx1+ system simulator SM+ are included.

 : The microcontroller operation simulation file to be used with system simulator SM+ for 78K0S/Kx1+ is included.

3.2 Internal Peripheral Functions to Be Used

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

- $V_{DD} < V_{LVI}$ detection: Low-voltage detector (LVI)
- PWM output function: 8-bit timer H1
- PWM output port (LED1): TOH1[Note]
- Output port (LED2): P21

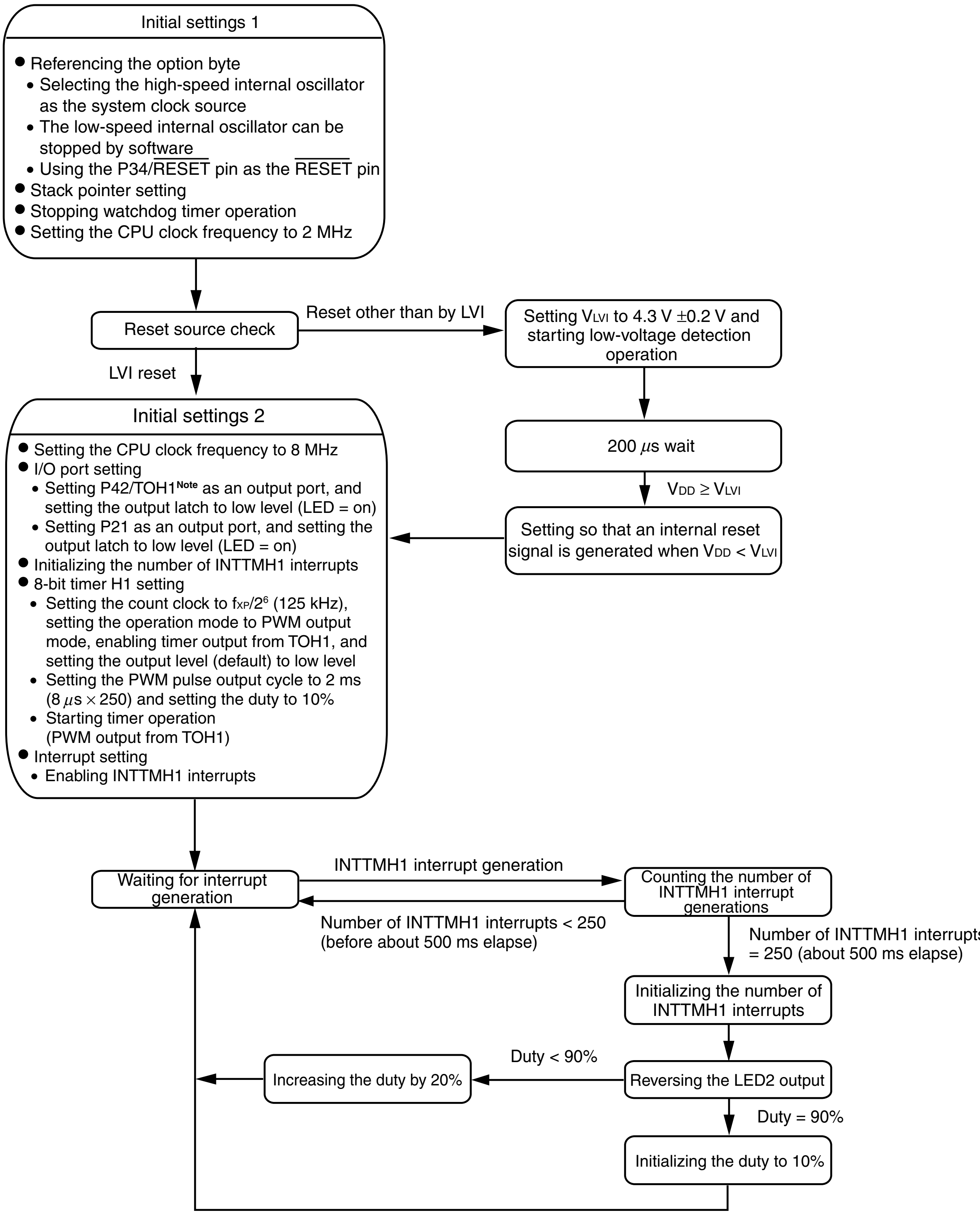
Note TOH1/P42: 78K0S/KA1+ and 78K0S/KB1+ microcontrollers
TOH1/ANI0/TI000/P20: 78K0S/KY1+ and 78K0S/KU1+ microcontrollers

3.3 Initial Settings and Operation Overview

In this sample program, initial settings including the setting of the low-voltage detection function, selection of the clock frequency, setting of the I/O ports, setting of 8-bit timer H1 (PWM output), and setting of interrupts are performed.

The brightness of LED1 is changed by controlling the PWM output duty of 8-bit timer H1, after completion of the initial settings. The duty is changed every 500 ms by using the 8-bit timer H1 interrupts (INTTMH1). The LED2 output is reversed when the duty is changed.

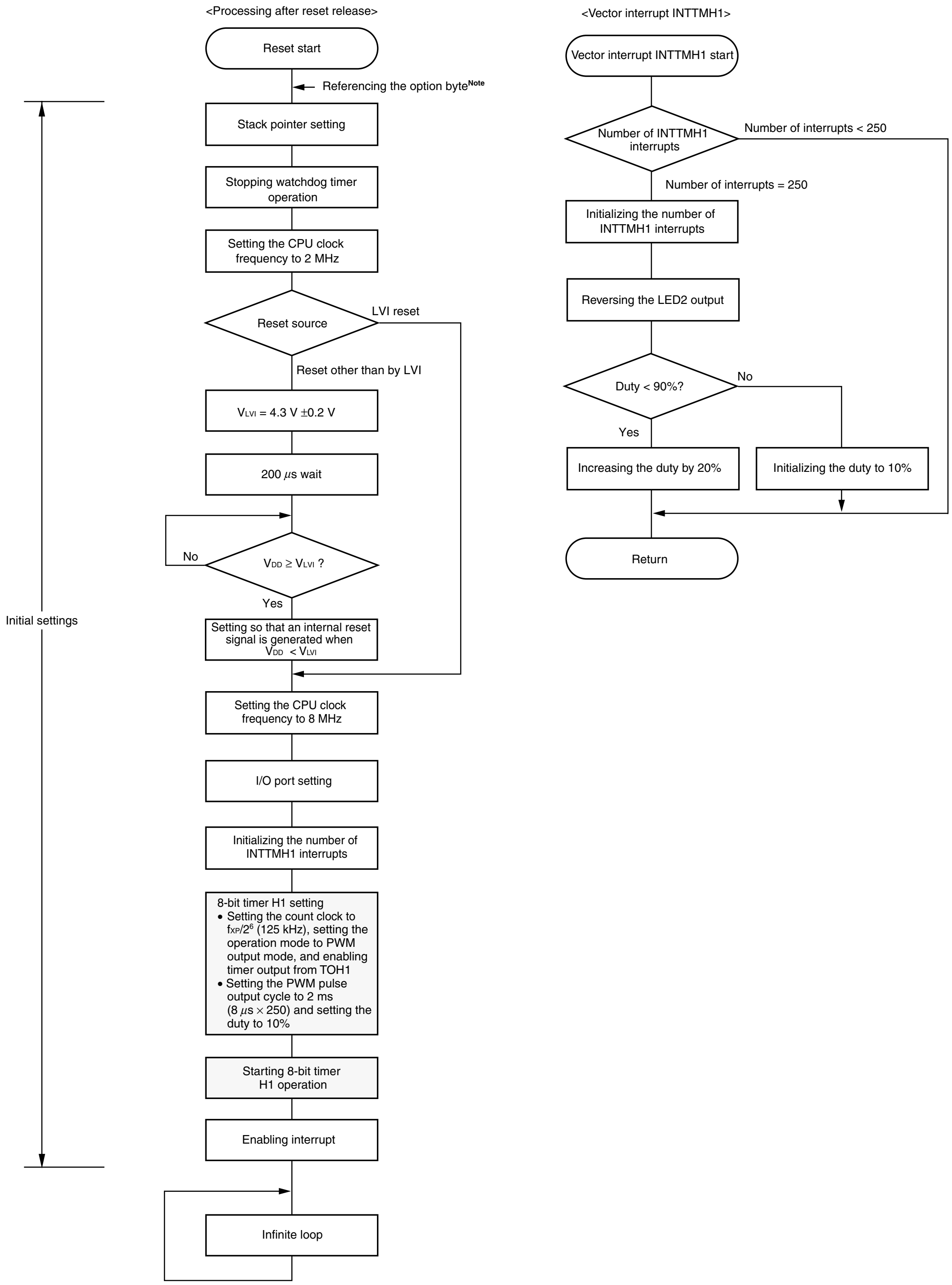
The details are described in the status transition diagram shown below.



Note TOH1/P42: 78K0S/KA1+ and 78K0S/KB1+ microcontrollers
TOH1/ANI0/TI000/P20: 78K0S/KY1+ and 78K0S/KU1+ microcontrollers

3.4 Flow Charts

The flow charts for the sample program are shown below.



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

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

CHAPTER 4 SETTING METHODS

This chapter describes the PWM output function of 8-bit timer H1.

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

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

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

4.1 Setting the PWM Output Function of 8-bit Timer H1

The following five types of registers are set when using 8-bit timer H1.

- 8-bit timer H mode register 1 (TMHMD1)
- 8-bit timer H compare register 01 (CMP01)
- Port mode register x (PMx)[Note]
- Port register x (Px)[Note]
- Port mode control register x (PMCx)[Note]

Note To use 8-bit timer H1 in PWM output mode, set it as follows.

	Px Register	PMx Register	PMCx Register
78K0S/KA1+ and 78K0S/KB1+ microcontrollers	P42 = 0	PM42 = 0	Setting not required
78K0S/KY1+ and 78K0S/KU1+ microcontrollers	P20 = 0	PM20 = 0	PMC20 = 0

(1) Setting regarding the operation mode of 8-bit timer H1

The operation mode is set, the count clock is selected, and operation is controlled for 8-bit timer H1 by using 8-bit timer H mode register 1 (TMHMD1).

Figure 4-1. Format of 8-bit Timer H Mode Register 1 (TMHMD1)

TMHMD1

TMHE1	CKS12	CKS11	CKS10	TMMD11	TMMD10	TOLEV1	TOEN1

Timer output (TOH1) control

TOEN1	
0	Disables output.
1	Enables output.

Timer output level control (default)

TOLEV1	
0	Low level
1	High level

Timer operation mode

TMMD11	TMMD10	
0	0	Interval timer mode
1	0	PWM output mode
Other than the above		Setting prohibited

Count clock selection

CKS12	CKS11	CKS10	
0	0	0	f_{XP}
0	0	1	$f_{XP}/2^2$
0	1	0	$f_{XP}/2^4$
0	1	1	$f_{XP}/2^6$
1	0	0	$f_{XP}/2^{12}$
1	0	1	$f_{RL}/2^7$
Other than the above			Setting prohibited

Enabling timer operation

TMHE1	
0	Stops operation (clears the counter to 0).
1	Enables operation.

Caution Setting the other bits of the TMHMD1 register is prohibited when TMHE1 is set to 1.

Remark f_{XP}: Oscillation frequency of the clock supplied to peripheral hardware
f_{RL}: Internal low-speed oscillation clock frequency

(2) Setting the PWM pulse output cycle and duty

8-bit timer H compare register 01 (CMP01) is used to set the PWM pulse output cycle, and 8-bit timer H compare register 11 (CMP11) is used to set the duty.

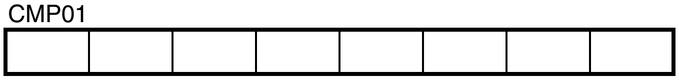
- PWM pulse output cycle = $(N + 1)/f_{CNT}$
- Duty = $(M + 1)/(N + 1)$

Remark N: CMP01 setting value
M: CMP11 setting value
f_{CNT}: Count clock frequency of 8-bit timer H1

Caution The CMP01 register setting value (N) and CMP11 register setting value (M) must be within the following ranges.
00H ≤ CMP11 (M) < CMP01 (N) ≤ FFH

Figure 4-2. Format of 8-bit Timer H Compare Register 01 (CMP01)

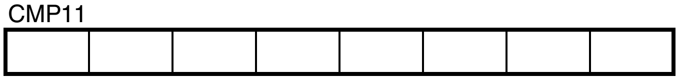
CMP01



Caution Rewriting the CMP01 register value during timer count operation is prohibited.

Figure 4-3. Format of 8-bit Timer H Compare Register 11 (CMP11)

CMP11



Cautions 1. The CMP11 register value can be rewritten during timer count operation. It takes 3 or more operation clocks (signal selected by the CKS12 to CKS10 bits of the TMHMD1 register), however, until the changed CMP11 register value is transferred to the register.

2. The CMP11 register must be set to start timer count operation (TMHE1 = 1) after timer count operation has been set to be stopped (TMHE1 = 0). (This must be re-set even if the value set to the CMP11 register is the same.)

3. When the CMP11 value is rewritten during timer operation, the compare value after the rewrite becomes valid at the timing when the count value matches the compare value before the rewrite. If the timing at which the count value and compare value match and writing from the CPU to CMP11 conflict, the compare value after the write becomes valid at the timing when the next count value matches the compare value before the write.

(3) TOH1 pin setting

To use 8-bit timer H1 in PWM output mode, set port register x (Px), port mode register x (PMx), and port mode control register x (PMCx) as follows.

	Px Register	PMx Register	PMCx Register
78K0S/KA1+ and 78K0S/KB1+ microcontrollers	P42 = 0	PM42 = 0	Setting not required
78K0S/KY1+ and 78K0S/KU1+ microcontrollers	P20 = 0	PM20 = 0	PMC20 = 0

[Example 1]

- Setting the operation mode of 8-bit timer H1 to PWM output mode, setting the count clock to $f_{XP}/2^6$ (f_{XP} = 8 MHz), enabling timer output (TOH1), and setting the output level (default) to low level
- Setting the PWM pulse output cycle to 2 ms, setting the duty to 10%, and starting timer operation (Same content as in the sample program)

TMHMD1

TMHE1	0	1	1	1	0	0	1

Timer output (TOH1) control

1	Enables output.

Timer output level control (default)

0	Low level

Timer operation mode

1	0	PWM output mode

Count clock selection

0	1	1	$f_{XP}/2^6$

Enabling timer operation

0	Stops operation (clears the counter to 0).
1	Enables operation.

CMP01 setting value (N): 249

- Count clock f_{CNT} = 8 MHz/2^6 = 0.125 MHz = 125 kHz
- PWM pulse output cycle 2 ms = (N + 1)/125 kHz

→ N = 2 ms × 125 kHz − 1 = 249

CMP11 setting value (M): 24

- 0.1 (= duty 10%) = (M + 1)/(249 + 1)

→ M = 0.1 × 250 − 1 = 24

TOH1 pin setting

- 78K0S/KA1+ and 78K0S/KB1+ microcontrollers: P42 = 0, PM42 = 0
- 78K0S/KY1+ and 78K0S/KU1+ microcontrollers: P20 = 0, PM20 = 0, PMC20 = 0

In the case of the 78K0S/KA1+ and 78K0S/KB1+ microcontrollers, timer operation is started by setting 1 to TMHE1 after setting “0” to P42, “0” to PM42, “00111001” to TMHMD1, “249” to CMP01, and “24” to CMP11.
In the case of the 78K0S/KY1+ and 78K0S/KU1+ microcontrollers, timer operation is started by setting 1 to TMHE1 after setting “0” to P20, “0” to PM20, “0” to PMC20, “00111001” to TMHMD1, “249” to CMP01, and “24” to CMP11.

• Assembly language (when using the 78K0S/KA1+ and 78K0S/KB1+ microcontrollers)

```
CLR1   P4.2
CLR1   PM4.2
MOV    TMHMD1,  #00111001B
MOV    CMP01,   #249
MOV    CMP11,   #24
SET1   TMHE1
```

• C language (when using the 78K0S/KA1+ and 78K0S/KB1+ microcontrollers)

```
P4.2 = 0;
PM4.2 = 0;
TMHMD1 = 0b00111001;
CMP01 = 249;
CMP11 = 24;
TMHE1 = 1;
```

[Example 2]
- Setting the operation mode of 8-bit timer H1 to PWM output mode, setting the count clock to f_{XP} (f_{XP} = 8 MHz), enabling timer output (TOH1), and setting the timer output level (default) to high level
- Setting the PWM pulse output cycle to 31.25 μs, setting the duty to 50%, and starting timer operation

TMHMD1

TMHE1	0	0	0	1	0	1	1

Timer output (TOH1) control

1	Enables output.

Timer output level control (default)

1	High level

Timer operation mode

1	0	PWM output mode

Count clock selection

0	0	0	f_{XP}

Enabling timer operation

0	Stops operation (clears the counter to 0).
1	Enables operation.

CMP01 setting value (N): 249

- Count clock f_{CNT} = 8 MHz
- PWM pulse output cycle 31.25 μs = (N + 1)/8 MHz

→ N = 31.25 μs × 8 MHz − 1 = 249

CMP11 setting value (M): 124

- 0.5 (= duty 50%) = (M + 1)/(249 + 1)

→ M = 0.5 × 250 − 1 = 124

TOH1 pin setting

- 78K0S/KA1+ and 78K0S/KB1+ microcontrollers: P42 = 0, PM42 = 0
- 78K0S/KY1+ and 78K0S/KU1+ microcontrollers: P20 = 0, PM20 = 0, PMC20 = 0

In the case of the 78K0S/KA1+ and 78K0S/KB1+ microcontrollers, timer operation is started by setting 1 to TMHE1 after setting "0" to P42, "0" to PM42, "00001011" to TMHMD1, "249" to CMP01, and "124" to CMP11.

In the case of the 78K0S/KY1+ and 78K0S/KU1+ microcontrollers, timer operation is started by setting 1 to TMHE1 after setting "0" to P20, "0" to PM20, "0" to PMC20, "00001011" to TMHMD1, "249" to CMP01, and "124" to CMP11.

• Assembly language (when using the 78K0S/KA1+ and 78K0S/KB1+ microcontrollers)

```
CLR1   P4.2
CLR1   PM4.2
MOV    TMHMD1, #00001011B
MOV    CMP01,  #249
MOV    CMP11,  #124
SET1   TMHE1
```

• C language (when using the 78K0S/KA1+ and 78K0S/KB1+ microcontrollers)

```
P4.2 = 0;
PM4.2 = 0;
TMHMD1 = 0b00001011;
CMP01 = 249;
CMP11 = 124;
TMHE1 = 1;
```

- Assembly language program example (same contents as in [**Example 1**] mentioned above and the sample program)

```
XMAIN  CSEG   UNIT
RESET_START:
                       •
                       •
                       •
       MOV    P4,     #00000000B          ; Set output latches of P40-P47 as low (P42:
turn on LED1)
       MOV    PM4,    #00000000B          ; Set P40-P47 as output mode
                       •
                       •
                       •
       MOV    TMHMD1,         #00111001B  ; Count clock = fxp/2^6 = 125 kHz,
                                          ; PWM mode, enable TOH1 output
       MOV    CMP01, #250-1               ; Initialize CMP01 (cycle: 2 ms)
       MOV    CMP11, #25-1                ; Initialize CMP11 (duty: 10%)
       SET1   TMHE1                       ; Start the timer operation

       MOV    IF0,    #00H                ; Clear invalid interrupt requests in advance
       CLR1   TMMKH1                      ; Unmask INTTMH1 interrupts

       EI                                 ; Enable vector interrupt

MAIN_LOOP:
       NOP
       BR     $MAIN_LOOP                  ; Go to the MAIN_LOOP
                       •
                       •
                       •
INTERRUPT_TMH1:
       DBNZ   CNT_TMH1, $END_INTTMH1      ; Branch if the number of INTTMH1 interrupts
< 250

       MOV    CNT_TMH1, #250              ; Initialize the number of interrupts
       XOR    P2,     #00000010B          ; Reverse LED2

       CMP    CMP11, #225-1               ; Compare the duty with that at 90%
       BC     $INC_CMP11                  ; Branch if the duty < 90%

       MOV    CMP11, #25-1                ; Initialize the duty to 10%
       BR     $END_INTTMH1                ; Branch to END_INTTMH1

INC_CMP11:
       ADD    CMP11, #50                  ; Increase the duty by 20%
                       •
                       •
                       •
```

Setting the P42 output latch and PM42 to 0
Setting the PWM pulse output cycle
Setting the 8-bit timer H1 operation mode, count clock, TOH1 output, and output level (default)
Starting timer operation
Setting the duty
Clearing the INTTMH1 interrupt request flag
Enabling INTTMH1 interrupt servicing
Starting interrupt servicing by INTTMH1 interrupt generation
The duty can be changed during 8-bit timer H1 operation

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

```
void hdwinit(void){
        unsigned char ucCnt200us;  /* 8-bit variable for 200 us wait */
                                •
                                •
                                •
        P4      = 0b00000000;           /* Set output latches of P40-P47 as low (P42: turn on
LED1) */
        PM4     = 0b00000000;           /* Set P40-P47 as output mode */
                                •
                                •
                                •
        TMHMD1 = 0b00111001;            /* Count clock = fxp/2^6 = 125 kHz, PWM mode, */
                                        /* Enable TOH1 output */
        CMP01 = 250-1;                  /* Initialize CMP01 (cycle: 2 ms) */
        CMP11 = 25-1;                   /* Initialize CMP11 (duty: 10%) */
        TMHE1 = 1;                      /* Start the timer operation */

        IF0    = 0x00;                  /* Clear invalid interrupt requests in advance */
        TMMKH1 = 0;                     /* Unmask INTTMH1 interrupts */

        return;
}

void main(void){

        EI();                           /* Enable vector interrupt */

        while (1){
                        NOP();
                        NOP();
        }
}

__interrupt void fn_inttmH1(){
                                •
                                •
                                •
                if (CMP11 >= 225-1){/* Processing when the duty is at least 90% */
                        CMP11 = 25-1;/* Initialize the duty to 10% */
                }
                else {
                        CMP11 += 50;  /* Increase the duty by 20% */
                                •
                                •
                                •
```

Setting the P42 output latch and PM42 to 0

Setting the 8-bit timer H1 operation mode, count clock, TOH1 output, and output level (default)

Setting the PWM pulse output cycle

Starting timer operation

Setting the duty

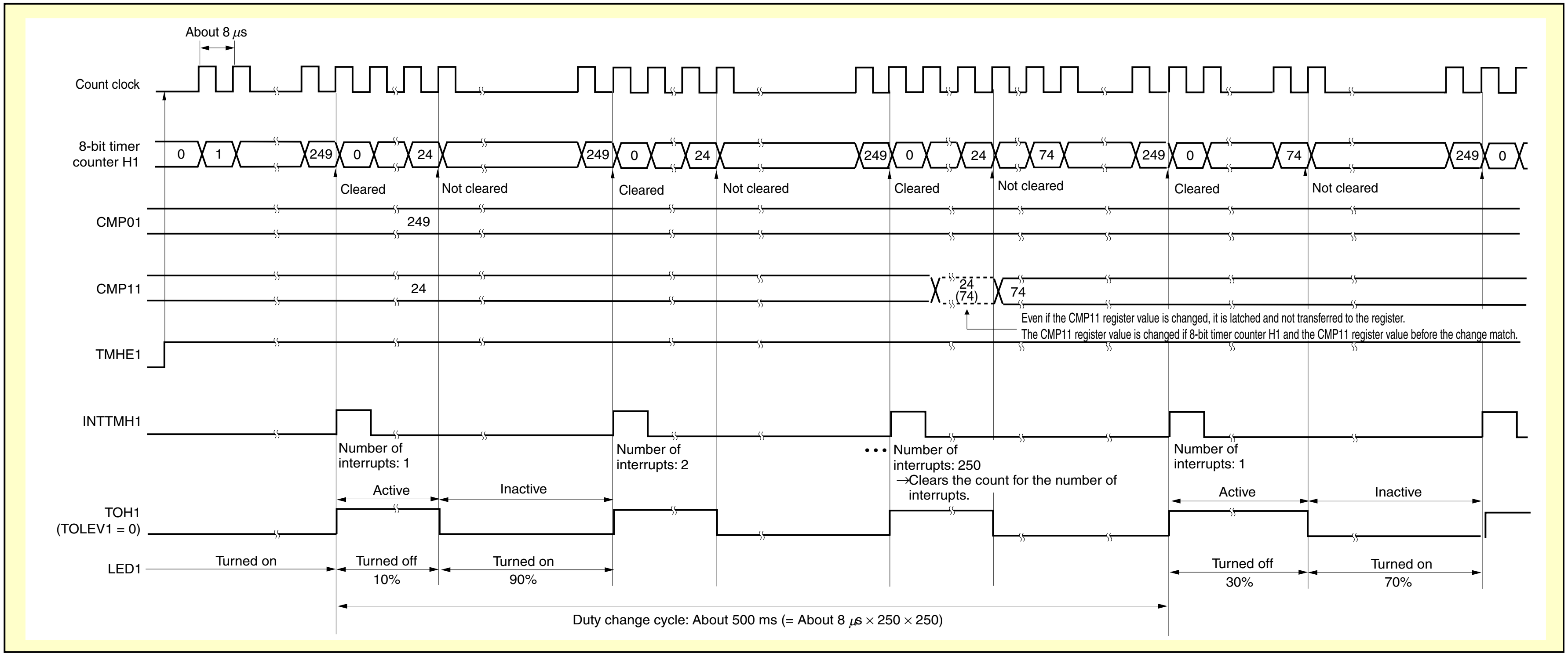
Clearing the INTTMH1 interrupt request flag

Enabling INTTMH1 interrupt servicing

Starting interrupt servicing by INTTMH1 interrupt generation

The duty can be changed during 8-bit timer H1 operation

Figure 4-4. Timing Chart Example of Changing the PWM Output Duty from 10% to 30% (the LED1 Brightness from 90% to 70%)



CHAPTER 5 OPERATION CHECK USING SYSTEM SIMULATOR SM+

This chapter describes how the sample program operates with system simulator SM+ for 78K0S/Kx1+, by using the assembly language file (source files + project file) that has been downloaded by selecting the icon.

<R> **Caution System simulator SM+ for 78K0S/Kx1+ is not supported with the 78K0S/KU1+ microcontroller (as of July 2008). The operation of the 78K0S/KU1+ microcontroller, therefore, cannot be checked by using system simulator SM+ for 78K0S/Kx1+.**

<R> 5.1 Building the Sample Program

To check the operation of the sample program by using system simulator SM+ for 78K0S/Kx1+ (hereinafter referred to as "SM+"), SM+ must be started after building the sample program. This section describes how to build a sample program by using the assembly language sample program (source program + project file) downloaded by clicking the icon. See the **78K0S/Kx1+ Sample Program Startup Guide Application Note** for how to build other downloaded programs.

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

[Column] Build errors

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

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

A RAM area of 118 bytes that has been secured as a fixed standard library area will be enabled for use when the [Using Fixed Area of Standard Library] check box is unchecked; however, the standard libraries (such as the getchar function and malloc function) will be disabled for use.

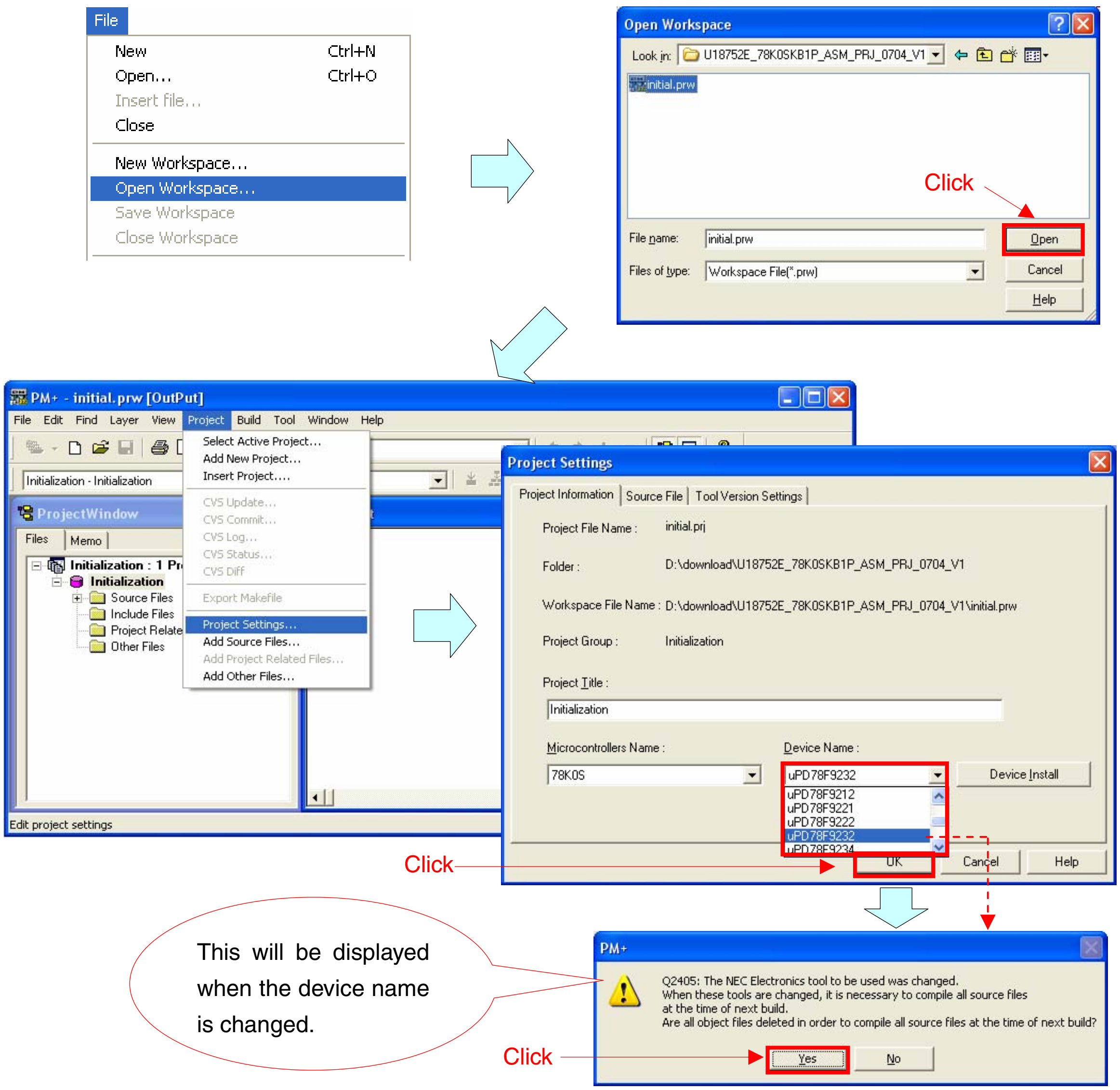
The [Using Fixed Area of Standard Library] check box is unchecked by default when the file that has been downloaded by clicking the icon is used in this sample program.

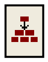
(1) Start PM+.

(2) Select “tmh1pwm.prw” by clicking [Open Workspace] from the [File] menu and click [Open]. A workspace into which the source file will be automatically read will be created.

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

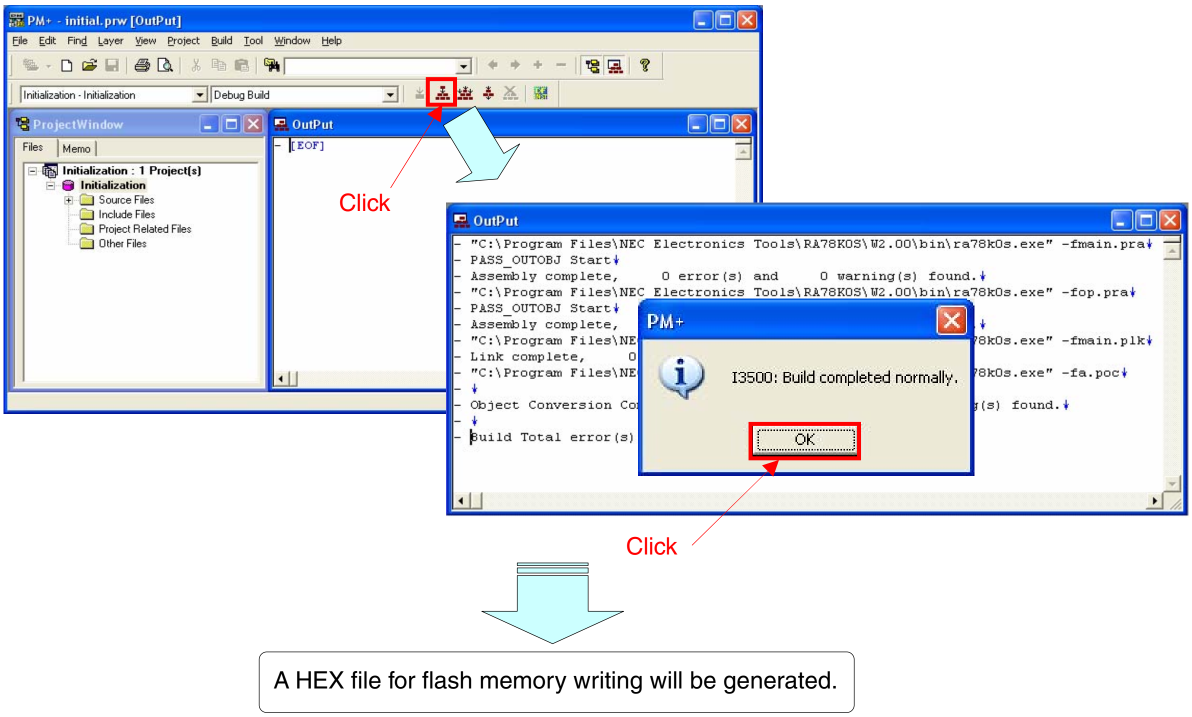
Remark Screenshots of the Sample Program (Initial Settings) LED Lighting Switch Control are shown below.



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

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

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



5.2 Operation with SM+

This section describes examples of checking the operation on the I/O panel window or timing chart window of SM+. For the details of how to operate SM+, refer to the **SM+ System Simulator Operation User’s Manual**.

<R> (1) When SM+ for 78K0S/Kx1+ W1.02 (“SM+” hereafter) is used in the environment of PM+ Ver. 6.30, SM+ cannot be selected as the debugger. In this case, start SM+ via method (a) or (b) described below, while keeping PM+ running after completing building a project.

(a) When starting SM+ in PM+

<1> Select [Register Ex-tool] from the [Tool] menu and register “SM+ for 78K0S/Kx1+”.

<2> Select [Ex-tool Bar] from the [View] menu and add the SM+ icon to the PM+ toolbar.

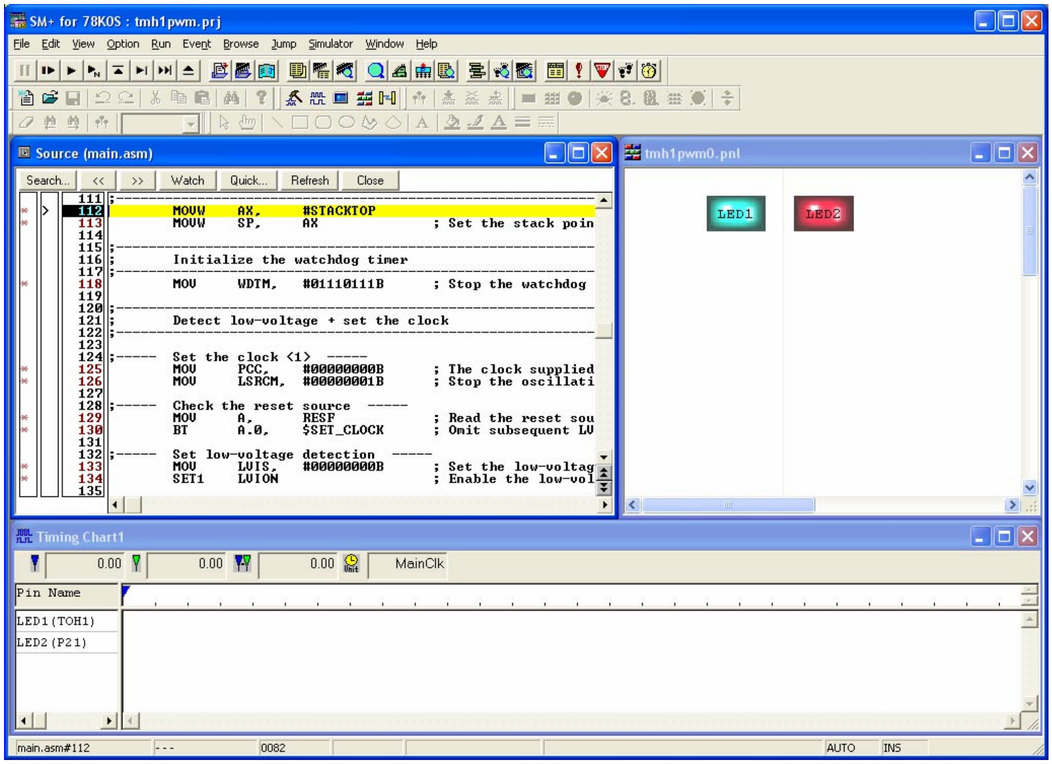
<3> Click the SM+ icon and start SM+.

(See the PM+ help for details on how to register external tools.)

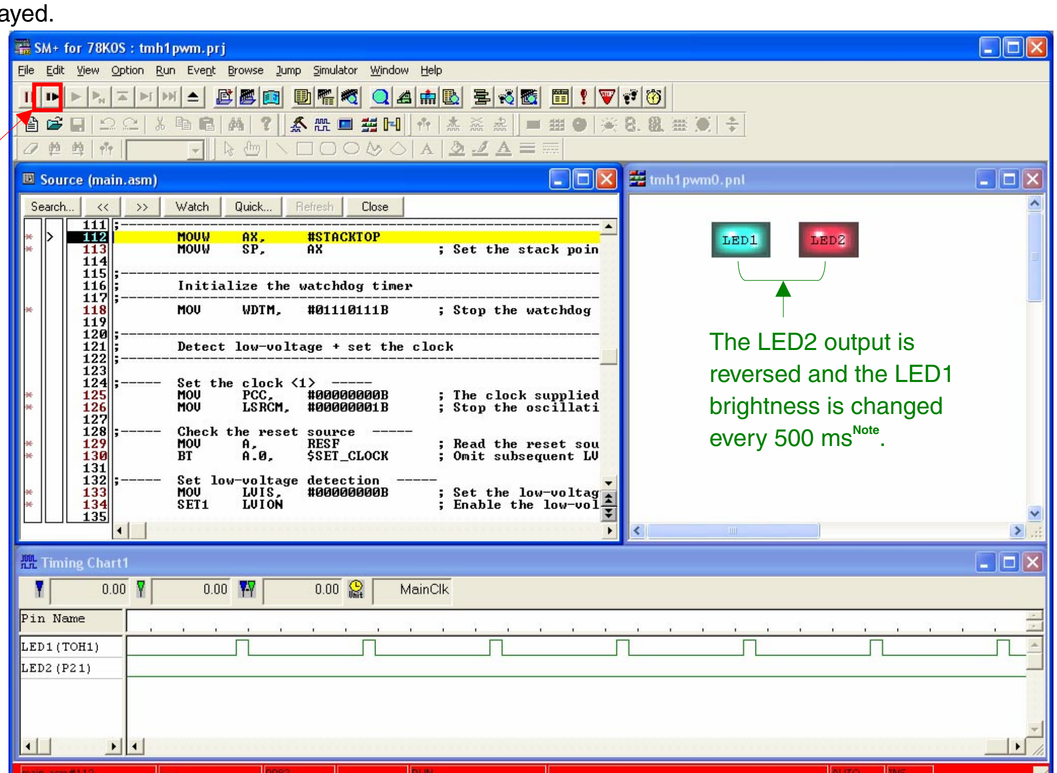
(b) When not starting SM+ in PM+

•Start SM+ from the Windows start menu.

(2) The following screen will be displayed when SM+ is started. (This is a sample screenshot of when an assembly language source file downloaded by clicking the  icon was used.)

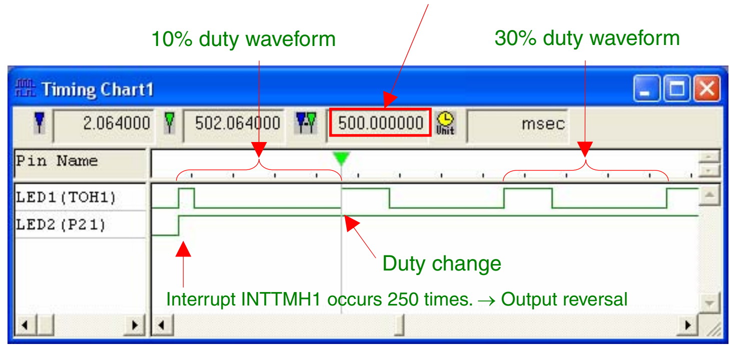
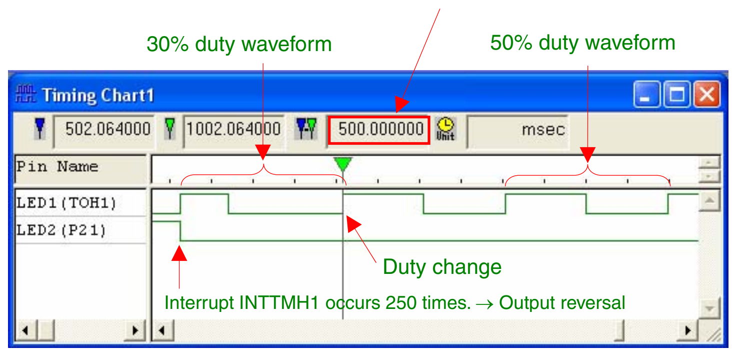
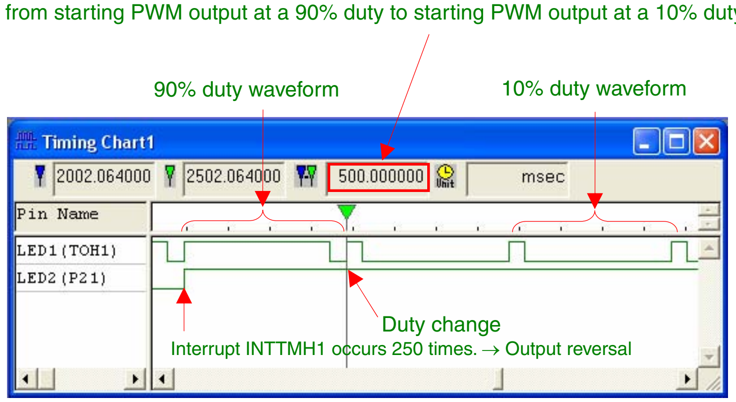


(3) Click  ([Restart] button). The program will be executed after the CPU is reset and the following screen will be displayed.



Note This may be different from the actual cycle, depending on the operation environment of the PC used.

(4) Check by viewing the waveforms in the timing chart window that the [LED2] output is reversed every time the PWM output duty ([LED1] brightness) changes during program execution.

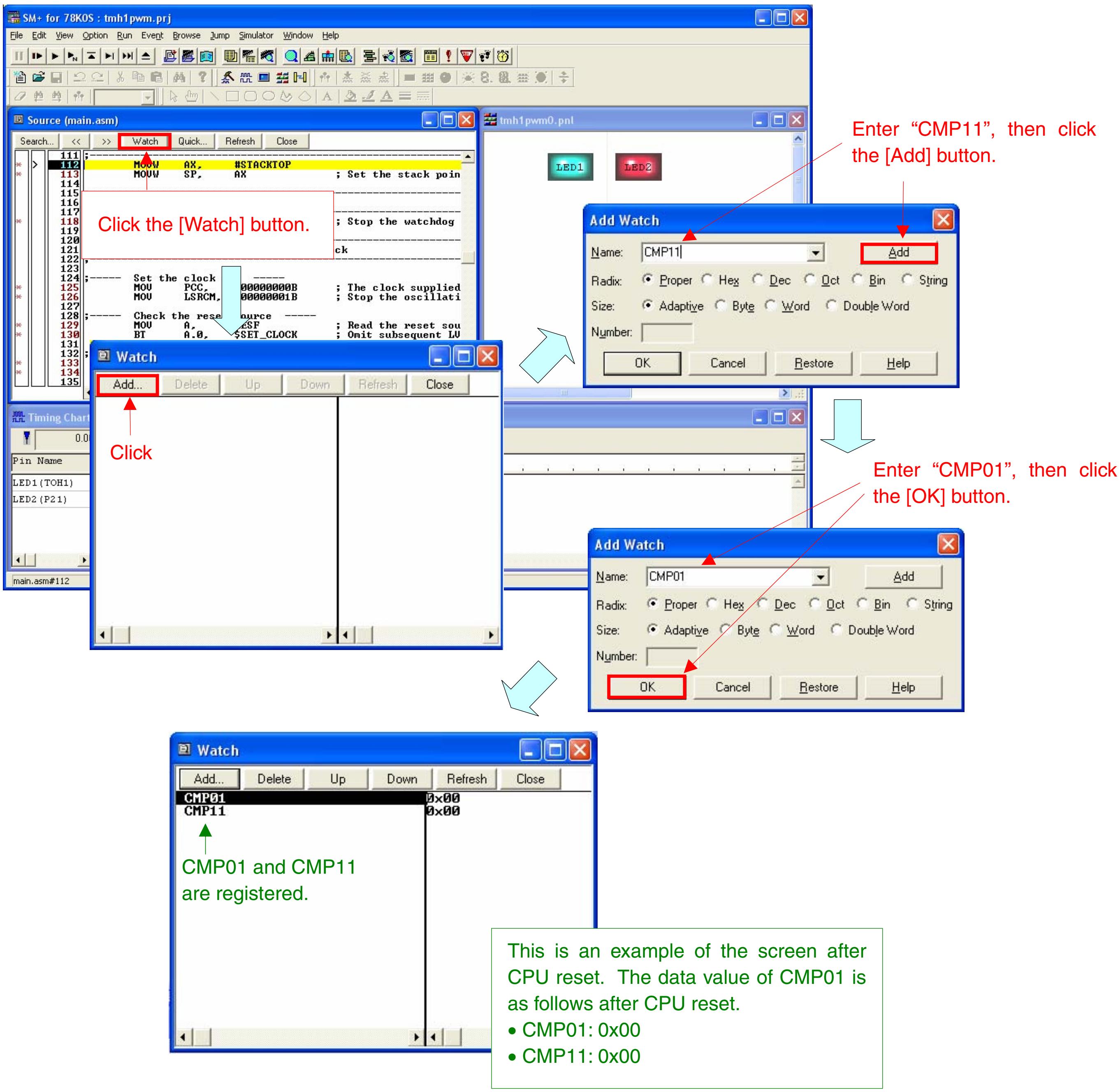
PWM Output Duty ([LED1] Brightness)	Timing Chart Window
Duty: 10% (brightness: 90%) ↓ Duty: 30% (brightness: 70%)	Time from starting PWM output at a 10% duty to starting PWM output at a 30% duty 
Duty: 30% (brightness: 70%) ↓ Duty: 50% (brightness: 50%)	Time from starting PWM output at a 30% duty to starting PWM output at a 50% duty 
• • •	• • •
Duty: 90% (brightness: 10%) ↓ Duty: 10% (brightness: 90%)	Time from starting PWM output at a 90% duty to starting PWM output at a 10% duty 

Note

Note The PWM output after a duty of 90% is repeated from a duty of 10%.

[Supplement] The changes in the data values of the CMP01 register and CMP11 register can be checked by using the SM+ watch function.

<1> Click the [Watch] button in the source window to open the [Watch] window.
<2> Click [Add] to open the [Add Watch] window. (At this time, the [Watch] window is kept opened.)
<3> Enter “CMP01” and “CMP11” in the [Name] field and click the [OK] button to register “CMP01” and “CMP11” in the [Watch] window and close the [Add Watch] window.



<4> Execute the program and check that the data values of CMP01 and CMP11 in the [Watch] window change.

PWM Output Duty ([LED1] Brightness)[Note]	Data Value in [Watch] Window
Duty: 10% (brightness: 90%)	CMP01: 0xF9 (249), CMP11: 0x18 (24)
Duty: 30% (brightness: 70%)	CMP01: 0xF9 (249), CMP11: 0x4A (74)
Duty: 50% (brightness: 50%)	CMP01: 0xF9 (249), CMP11: 0x7C (124)
Duty: 70% (brightness: 30%)	CMP01: 0xF9 (249), CMP11: 0xAE (174)
Duty: 90% (brightness: 10%)	CMP01: 0xF9 (249), CMP11: 0xE0 (224)

Note The PWM output after a duty of 90% is repeated from a duty of 10%.

CHAPTER 6 RELATED DOCUMENTS

Document Name		Japanese/English
78K0S/KU1+ User's Manual		PDF
78K0S/KY1+ User's Manual		PDF
78K0S/KA1+ User's Manual		PDF
78K0S/KB1+ User's Manual		PDF
78K/0S Series Instructions User's Manual		PDF
RA78K0S Assembler Package User's Manual	Language	PDF
	Operation	PDF
CC78K0S C Compiler User's Manual	Language	PDF
	Operation	PDF
PM+ Project Manager User's Manual		PDF
SM+ System Simulator Operation User's Manual		PDF
<R> Flash Programming Manual (Basic) MINICUBE2 version	78K0S/KU1+	PDF
	78K0S/KY1+	PDF
	78K0S/KA1+	PDF
	78K0S/KB1+	PDF
78K0S/Kx1+ Application Note	Sample Program Startup Guide	PDF
	Sample Program (Initial Settings) LED Lighting Switch Control	PDF
	Sample Program (Interrupt) External Interrupt Generated by Switch Input	PDF
	Sample Program (Low-Voltage Detection) Reset Generation During Detection at Less than 2.7 V	PDF
	Sample Program (8-bit Timer H1) Interval Timer	PDF

APPENDIX A PROGRAM LIST

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

● main.asm (Assembly language version)

```
;*****************************************************************************
;
;     NEC Electronics     78K0S/KB1+
;
;*****************************************************************************
;     78K0S/KB1+  Sample program
;*****************************************************************************
;     8-bit timer H1 (PWM output)
;*****************************************************************************
;<<History>>
;     2007.7.--   Release
;*****************************************************************************
;
;<<Overview>>
;
;This sample program presents an example of using the PWM output function of
;8-bit timer H1.  The LED1 brightness is controlled through the PWM output
;duty by connecting the timer output (TOH1 pin) of 8-bit timer H1 to LED1.
;The duty is changed at a cycle of 500 ms by using timer H1 interrupts and
;LED2 output is reversed simultaneously.
;
;
;  <Principal setting contents>
;
;  - Stop the watchdog timer operation
;  - Set the low-voltage detection voltage (VLVI) to 4.3 V +-0.2 V
;  - Generate an internal reset signal (low-voltage detector) when VDD < VLVI
after VDD >= VLVI
;  - Set the CPU clock to 8 MHz
;  - Set the clock supplied to the peripheral hardware to 8 MHz
;
;
;  <8-bit timer H1 settings>
;  - Set to the PWM mode
;  - Enable timer output of the TOH1 pin
;  - Count clock = fxp/2^6 (125 kHz)
;  - Timer cycle = 2 ms (8[us/clk] x 250[count] = 2[ms])
;
;
;  <PWM output duty and LEDs>
;
```

```
;   - LED2 output is reversed simultaneously with the duty that is changed
every 500 ms in the following order.
;    +-----------------+-----------------------------+
;    | PWM output duty | 10% | 30% | 50% | 70% | 90% | (Hereafter, repeated
from 10%)
;    +-----------------+-----------------------------+
;    | LED1 brightness  | 90% | 70% | 50% | 30%  | 10% |
;    +-----------------+-----------------------------+
;    # PWM output is high active and LED1 is low active; therefore, the LED
;      brightness = 100 - duty factor.
;
;
;<<I/O port settings>>
;
;  Input: -
;  Output: P00-P03, P20-P23, P30-P33, P40-P47, P120-P123, P130
;  # All unused ports are set as the output mode.
;
;******************************************************************************


;==============================================================================
;
;     Vector table
;
;==============================================================================
XVCT  CSEG  AT    0000H
      DW    RESET_START       ;(00) RESET
      DW    RESET_START       ;(02) --
      DW    RESET_START       ;(04) --
      DW    RESET_START       ;(06) INTLVI
      DW    RESET_START       ;(08) INTP0
      DW    RESET_START       ;(0A) INTP1
      DW    INTERRUPT_TMH1    ;(0C) INTTMH1
      DW    RESET_START       ;(0E) INTTM000
      DW    RESET_START       ;(10) INTTM010
      DW    RESET_START       ;(12) INTAD
      DW    RESET_START       ;(14) --
      DW    RESET_START       ;(16) INTP2
      DW    RESET_START       ;(18) INTP3
      DW    RESET_START       ;(1A) INTTM80
      DW    RESET_START       ;(1C) INTSRE6
      DW    RESET_START       ;(1E) INTSR6
      DW    RESET_START       ;(20) INTST6

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

```
;
;==============================================================================
XRAM  DSEG  SADDR
CNT_TMH1:   DS    1           ; For counting INTTMH1 interrupt

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

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

;------------------------------------------------------------------------------
;     Initialize the watchdog timer
;------------------------------------------------------------------------------
      MOV   WDTM, #01110111B  ; Stop the watchdog timer operation

;------------------------------------------------------------------------------
;     Detect low-voltage + set the clock
;------------------------------------------------------------------------------

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

;-----  Check the reset source  -----
      MOV   A,    RESF        ; Read the reset source
      BT    A.0,  $SET_CLOCK  ; Omit subsequent LVI-related processing and go
to SET_CLOCK during LVI reset
```

```
;-----  Set low-voltage detection  -----
      MOV   LVIS, #00000000B  ; Set the low-voltage detection level (VLVI) to
4.3 V +-0.2 V
      SET1  LVION             ; Enable the low-voltage detector operation

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

;-----  VDD >= VLVI wait processing  -----
WAIT_LVI:
      NOP
      BT    LVIF, $WAIT_LVI   ; Branch if VDD < VLVI

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

;-----  Set the clock <2>  -----
SET_CLOCK:
      MOV   PPCC, #00000000B  ; The clock supplied to the peripheral hardware
(fxp) = fx (= 8 MHz)
                              ;  -> The clock supplied to the CPU (fcpu) = fxp
= 8 MHz

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

;-----------------------------------------------------------------------------
;     Initialize the port 2
;-----------------------------------------------------------------------------
      MOV   P2,   #00000000B  ;  Set  output  latches  of  P20-P23  as  low  (P21:
turn on LED2)
      MOV   PM2,  #11110000B  ; Set P20-P23 as output mode

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

;-----------------------------------------------------------------------------
;     Initialize the port 4
;-----------------------------------------------------------------------------
```

```
      MOV   P4,   #00000000B  ; Set output latches of P40-P47 as low (P42:
turn on LED1)
      MOV   PM4,  #00000000B  ; Set P40-P47 as output mode

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

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

;-----------------------------------------------------------------------------
;     Initialize the RAM
;-----------------------------------------------------------------------------
      MOV   CNT_TMH1, #250    ; Initialize the number of INTTMH1 interrupts

;-----------------------------------------------------------------------------
;     Set 8-bit timer H1
;-----------------------------------------------------------------------------
      MOV   TMHMD1,     #00111001B  ; Count clock = fxp/2^6 = 125 kHz, PWM
mode, enable TOH1 output
      MOV   CMP01,      #250-1; Initialize CMP01 (cycle: 2 ms)
      MOV   CMP11,      #25-1 ; Initialize CMP11 (duty: 10%)
      SET1  TMHE1             ; Start the timer operation

;-----------------------------------------------------------------------------
;     Set the interrupt
;-----------------------------------------------------------------------------
      MOV   IF0,  #00H        ; Clear invalid interrupt requests in advance
      CLR1  TMMKH1            ; Unmask INTTMH1 interrupts

      EI                      ; Enable vector interrupt

;*****************************************************************************
;
;     Main loop
;
;*****************************************************************************
MAIN_LOOP:
      NOP
      BR    $MAIN_LOOP        ; Go to the MAIN_LOOP

;*****************************************************************************
;
```

```
;       Interrupt INTTMH1
;
;*****************************************************************************
INTERRUPT_TMH1:
        DBNZ  CNT_TMH1, $END_INTTMH1  ;  Branch  if  the  number  of  INTTMH1
interrupts < 250

        MOV   CNT_TMH1, #250          ; Initialize the number of interrupts
        XOR   P2,   #00000010B        ; Reverse LED2

        CMP   CMP11,      #225-1      ; Compare the duty with that at 90%
        BC    $INC_CMP11              ; Branch if the duty < 90%

        MOV   CMP11,      #25-1       ; Initialize the duty to 10%
        BR    $END_INTTMH1            ; Branch to END_INTTMH1

INC_CMP11:
        ADD   CMP11,      #50         ; Increase the duty by 20%

END_INTTMH1:
        RETI                          ; Return from interrupt servicing

end
```

● main.c (C language version)

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

      NEC Electronics     78K0S/KB1+

******************************************************************************
      78K0S/KB1+  Sample program
******************************************************************************
      8-bit timer H1 (PWM output)
******************************************************************************
<<History>>
      2007.7.--   Release
******************************************************************************

<<Overview>>

This sample program presents an example of using the PWM output function of 8-
bit timer H1.  The LED1 brightness is controlled through the PWM output duty
by connecting the timer output (TOH1 pin) of 8-bit timer H1 to LED1.  The duty
is changed at a cycle of 500 ms by using timer H1 interrupts and LED2 output
is reversed simultaneously.


  <Principal setting contents>

  - Stop the watchdog timer operation
  - Set the low-voltage detection voltage (VLVI) to 4.3 V +-0.2 V
  - Generate an internal reset signal (low-voltage detector) when VDD < VLVI
after VDD >= VLVI
  - Set the CPU clock to 8 MHz
  - Set the clock supplied to the peripheral hardware to 8 MHz


  <8-bit timer H1 settings>
  - Set to the PWM mode
  - Enable timer output of the TOH1 pin
  - Count clock = fxp/2^6 (125 kHz)
  - Timer cycle = 2 ms (8[us/clk] x 250[count] = 2[ms])


  <PWM output duty and LEDs>

  - LED2 output is reversed simultaneously with the duty that is changed every
500 ms in the following order.
    +-----------------+----------------------------+
    | PWM output duty | 10% | 30% | 50% | 70% | 90% |   (Hereafter,   repeated
from 10%)
```

```
    +----------------+----------------------------+
    | LED1 brightness | 90% | 70% | 50% | 30% | 10% |
    +----------------+----------------------------+
    # PWM output is high active and LED1 is low active; therefore, the LED
      brightness = 100 - duty factor.


<<I/O port settings>>

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

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


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

      Preprocessing directive (#pragma)

============================================================================*/
#pragma     SFR                  /* SFR names can be described at the C
source level */
#pragma     EI                   /* EI instructions can be described at the
C source level */
#pragma     NOP                  /* NOP instructions can be described at
the C source level */
#pragma interrupt INTTMH1 fn_inttmH1      /*       Interrupt       function
declaration:INTTMH1 */

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

      Define the global variables

============================================================================*/
sreg unsigned char ucTMH1cnt = 250; /* 8-bit variable for counting the number
of INTTMH1 interrupts */

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

      Initialization after RESET

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

/*----------------------------------------------------------------------------
      Initialize the watchdog timer + detect low-voltage + set the clock
```

```
-----------------------------------------------------------------------------*/
      /* Initialize the watchdog timer */
      WDTM  = 0b01110111;           /* Stop the watchdog timer operation */

      /* Set the clock <1> */
      PCC   = 0b00000000;           /* The clock supplied to the CPU (fcpu) =
fxp (= fx/4 = 2 MHz) */
      LSRCM = 0b00000001;           /* Stop the oscillation of the low-speed
internal oscillator */

      /* Check the reset source */
      if (!(RESF & 0b00000001)){    /* Omit subsequent LVI-related processing
during LVI reset */

            /* Set low-voltage detection */
            LVIS  = 0b00000000;     /* Set the low-voltage detection level
(VLVI) to 4.3 V +-0.2 V */
            LVION = 1;              /* Enable the low-voltage detector
operation */

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

            while (LVIF){           /* Wait for VDD >= VLVI */
                  NOP();
            }

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

      /* Set the clock <2> */
      PPCC  = 0b00000000;           /* The clock supplied to the peripheral
hardware (fxp) = fx (= 8 MHz)
                                       -> The clock supplied to the CPU (fcpu)
= fxp = 8 MHz */

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

/*-----------------------------------------------------------------------------
      Initialize the port 2
-----------------------------------------------------------------------------*/
```

```
      P2    = 0b00000000;           /* Set output latches of P20-P23 as low
(P21: turn on LED2) */
      PM2   = 0b11110000;           /* Set P20-P23 as output mode */

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

/*------------------------------------------------------------------------------
      Initialize the port 4
------------------------------------------------------------------------------*/
      P4    = 0b00000000;           /* Set output latches of P40-P47 as low
(P42: turn on LED1) */
      PM4   = 0b00000000;           /* Set P40-P47 as output mode */

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

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

/*------------------------------------------------------------------------------
      Set 8-bit timer H1
------------------------------------------------------------------------------*/
      TMHMD1 = 0b00111001;          /* Count clock = fxp/2^6 = 125 kHz, PWM
mode, enable TOH1 output */
      CMP01 = 250-1;                /* Initialize CMP01 (cycle: 2 ms) */
      CMP11 = 25-1;                 /* Initialize CMP11 (duty: 10%) */
      TMHE1 = 1;                    /* Start the timer operation */

/*------------------------------------------------------------------------------
      Set the interrupt
------------------------------------------------------------------------------*/
      IF0   = 0x00;                 /* Clear invalid interrupt requests in
advance */
      TMMKH1 = 0;                   /* Unmask INTTMH1 interrupts */

      return;
}
```

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

      Main loop

******************************************************************************/
void main(void){

      EI();                        /* Enable vector interrupt */

      while (1){
            NOP();
            NOP();
      }
}

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

      Interrupt INTTMH1

******************************************************************************/
__interrupt void fn_inttmH1(){

      if (--ucTMH1cnt == 0){       /* Processing when the number of INTTMH1
interrupts is 250 */

            ucTMH1cnt = 250;       /* Initialize the number of interrupts */
            P2 ^= 0b00000010;      /* Reverse LED2 */

            if (CMP11 >= 225-1){   /* Processing when the duty is at least
90% */
                  CMP11 = 25-1;    /* Initialize the duty to 10% */
            }
            else {
                  CMP11 += 50;     /* Increase the duty by 20% */
            }
      }

      return;
}
```

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

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

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

end
```

APPENDIX B REVISION HISTORY

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

Edition	Date Published	Page	Revision
1st edition	November 2007	–	–
2nd edition	September 2008	p.20	CHAPTER 5 OPERATION CHECK USING SYSTEM SIMULATOR SM+ • Modification of description in Caution ((as of July 2007) → (as of July 2008))
		pp.20 to 22	Modification of 5.1 Building the Sample Program
		p.22	5.2 Operation with SM+ • Addition of (1)
		p.27	CHAPTER 6 RELATED DOCUMENTS • Addition of Flash Programming Manual (Basic) MINICUBE2 version

For further information, please contact:

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

[America]

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

[Europe]

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

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

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

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

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

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

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

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

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

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

[Asia & Oceania]

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

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

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

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

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

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

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

G0706