To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

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

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

M16C/64 Group

Watchdog Timer

1. Abstract

The watchdog timer is the facility to detect program runaway.
The watchdog timer has a 15-bit counter and provides a facility to enable or disable count source protection mode.
Table 1 shows specifications of the watchdog timer.

Table 1. Watchdog Timer Specifications

Item	When count source protection mode is disabled	When count source protection mode is enabled
Count source	CPU clock	125 kHz on-chip oscillator clock
Count operation	Down-count	
Count start condition	• If WDTON bit at OFS1 address = 0 After reset, the watchdog timer automatically starts counting. • If WDTON bit at OFS1 address = 1 The watchdog timer starts counting by a write to the WDTS register.	
Count stop condition	Stop mode, wait mode or hold state	None
Operation on underflow	• If PM1 register PM12 bit = 0 Watchdog timer interrupt • If PM1 register PM12 bit = 1 Watchdog timer reset	Watchdog timer reset

The watchdog timer cycle differs depending on the CPU clock, prescaler divide ratio and whether count source protection mode is enabled. The watchdog timer cycles varying with each setting are listed in Table 2.

Table 2. Watchdog Timer Cycles

CSPR register CSPRO bit Note 1, Note 2	CM0 register CM07 bit	WDC register WDC7 bit	Watchdog timer cycle
0 (count source protection mode disabled)	0 (CPU clock is the main clock, PLL clock or 125 kHz on-chip oscillator)	0 (divided by 16)	$\frac{\text{16 (prescaler divide ratio)} \times \text{32768 (WDT count value)}}{\text{CPU clock}}$
		1 (divided by 128)	$\frac{\text{128 (prescaler divide ratio)} \times \text{32768 (WDT count value)}}{\text{CPU clock}}$
	1 (CPU clock is a sub-clock)	—	$\frac{\text{2 (prescaler divide ratio)} \times \text{32768 (WDT count value)}}{\text{CPU clock}}$
1 (count source protection mode enabled)	—	—	$\frac{\text{4096 (WDT count value)}}{\text{125 kHz on-chip oscillator}}$

— : Don't care (can be 1 or 0).
WDT : Watchdog timer
Note 1 : Unless the CSPROINI bit at the OFS1 address is 0 (after-reset count source protection mode enabled), the CSPRO bit is not set to 1 (count source protection mode enabled) after reset.
Note 2 : To set the CSPRO bit to 1, write 0 and then write 1 in succession in a program. The CSPRO bit cannot be cleared to 0 (count source protection mode disabled) in a program.

2. Introduction

The application example described in this document applies to the following MCU:

- Microcomputers: M16C/64 group

This application note can be used with other M16C Family MCUs which have the same special function registers (SFRs) as the above group. Check the manual for any modifications to functions. Careful evaluation is recommended before using the program described in this application note.

3. Application Example

In the explanation here, we show how to generate a watchdog timer interrupt or a watchdog timer reset using the CPU clock as the count source for the watchdog timer.

3.1 Description of the Application Example

(1) Write 00h and then write FFh to the WDTR register in succession. The watchdog timer is initialized to 7FFFh Note 1.

(2) The watchdog timer starts counting by a write to the WDTS register.

(3) Write to the WDTR register again while the count is in progress. The watchdog timer is initialized to 7FFFh Note 1, from which it continues counting.

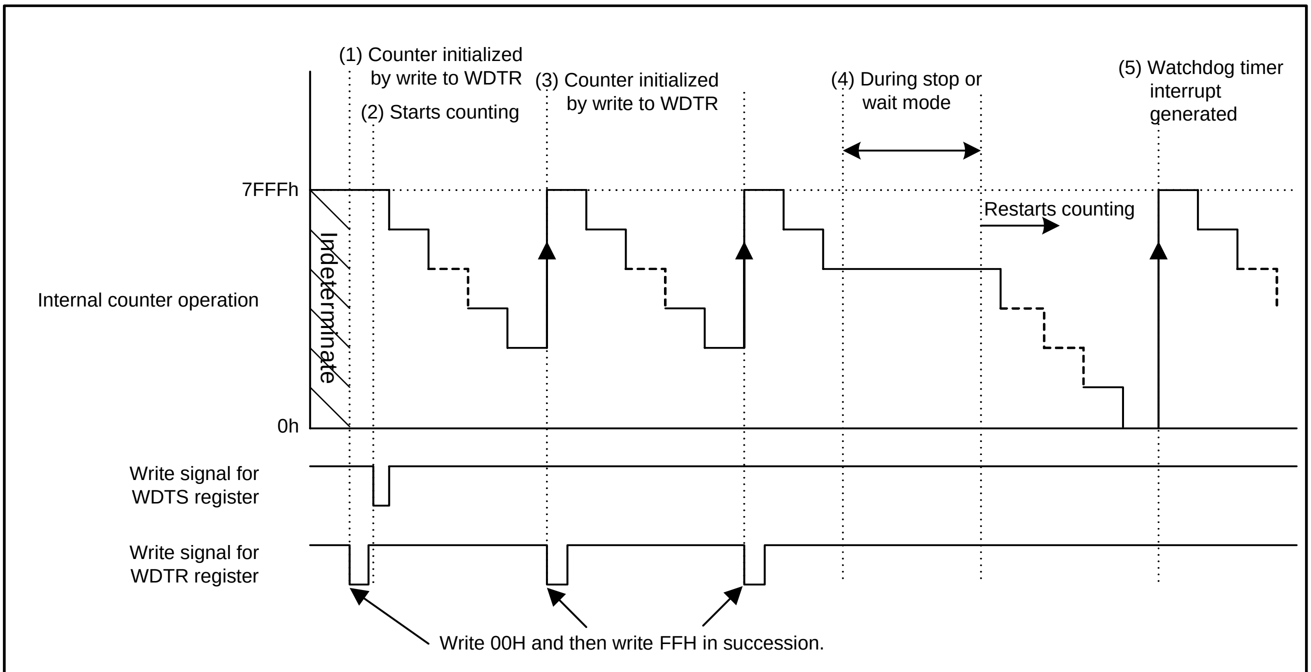
(4) When in wait mode, stop mode or hold state, the watchdog timer stops counting, retaining the last count it had before stopping. When placed out of wait mode or stop mode, it restarts counting from the count value it retained Note 2.

(5) Where a watchdog timer interrupt is used: When the watchdog timer underflows upon reaching the terminal count, a watchdog timer interrupt is generated. In this case, because the watchdog timer is not initialized, perform the write to the WDTR register described in (1) within an interrupt handler. The watchdog timer continues counting.
Where a watchdog timer reset is used: When the watchdog timer underflows upon reaching the terminal count, the pins and the CPU and SFR are initialized, running the program from the address indicated by the reset vector.

Note 1: If the CSPR register CSPRO bit = 1 (count source protection mode enabled), the watchdog timer is initialized to 0FFFh.

Note 2: If the CSPR register CSPRO bit = 1 (count source protection mode enabled), the watchdog timer does not stop even when it is in wait mode, stop mode or hold state.

Figure 1 shows an operation timing diagram.


(1) Counter initialized by write to WDTR
(2) Starts counting
(3) Counter initialized by write to WDTR
(4) During stop or wait mode
(5) Watchdog timer interrupt generated
7FFFh
Indeterminate
Internal counter operation
Restarts counting
0h
Write signal for WDTS register
Write signal for WDTR register
Write 00H and then write FFH in succession.

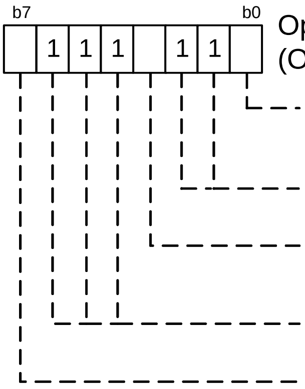
This diagram applies to the following register settings.
PM1 register PM12 bit = 0 (watchdog timer interrupt)
CSPR register CSPRO bit = 0 (count source protection mode disabled)

Figure 1. Watchdog Timer Operation Example

4. How to Set Up

The following shows how to set up the registers to accomplish the operation described in 3, “Application Example.” For details about each register, see the hardware manual of the M16C/64 group.

(1) Setting the optional feature select address



Optional feature select address (OFS1)

The OFS1 address is located in the flash memory (address FFFFFh). Write to this address along with the program.

If the block that includes the OFS1 address is erased, the OFS1 address becomes FFh.

WDTON Watchdog timer start select bit
0: Automatically starts the watchdog timer after reset
1: Keeps the watchdog timer idle after reset

Reserved bits
Set these bits to 1.

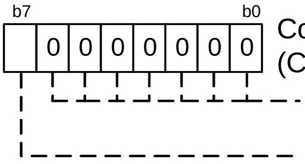
ROMCP1 ROM code protect bit
0: Enables ROM code protect
1: Disables ROM code protect

Reserved bits
Set these bits to 1.

CSPROINI After-reset count source protection mode select bit
0: Enables after-reset count source protection mode
1: Disables after-reset count source protection mode

When setting the CSPROINI bit to 0, be sure to set the WDTON bit to 0 also.

(2) Setting the count source protection mode register



Count source protection mode register (CSPR)

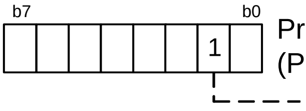
If the CSPROINI bit at the OFS1 address is cleared by writing 0, the value of this register after reset is ‘10000000b.’

Reserved bits
Set these bits to 0.

CSPRO Count source protection mode select bit
0: Disables count source protection mode
1: Enables count source protection mode

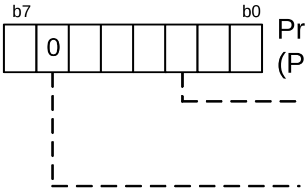
To set the CSPRO bit to 1, write 0 and then write 1 in succession. This bit cannot be cleared to 0 in a program.

(3) Setting processor mode register 1



Protect register (PRCR)

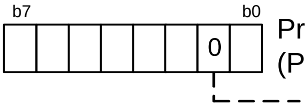
PRC1 Protect bit 1
PM0, PM1, PM2, TB2SC, INVC0, INVC1 register write enable
0: Disables write
1: Enables write



Processor mode register 1 (PM1)

PM12 Watchdog timer facility select bit
0: Watchdog timer interrupt
1: Watchdog timer reset

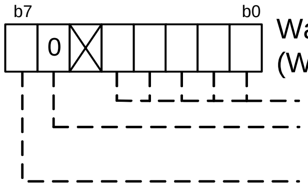
Reserved bit
Set this bit to 0.



Protect register (PRCR)

PRC1 Protect bit 1
PM0, PM1, PM2, TB2SC, INVC0, INVC1 register write enable
0: Disables write
1: Enables write

(4) Setting the watchdog timer control register



Watchdog timer control register (WDC)

Watchdog timer high-order bits

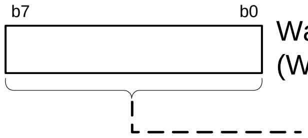
Reserved bit
Set this bit to 0.

WDC7 Prescaler select bit
0: Divide by 16
1: Divide by 128

If the CSPRO bit = 1 (count source protection mode enabled), the on-chip oscillator is selected for the count source, in which case the clock frequency division by the prescaler is not performed no matter how WDC7 is set.

If the CSPRO bit = 0 (count source protection mode disabled) and the selected CPU clock is a sub-clock, a division by 2 is assumed for the clock frequency division by the prescaler no matter how WDC7 is set.

(5) Setting the watchdog timer reset register

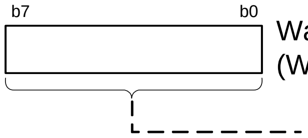


Watchdog timer reset register (WDTR)

Write 00h and then write FFh to this register in succession. The watchdog timer is initialized.
Watchdog timer initial value
- When count source protection mode is disabled: 7FFFh
- When count source protection mode is enabled: 0FFFh

Be sure that no interrupt or DMA transfer occurs between a write of 00h and a write of FFh.
After a watchdog timer interrupt is generated, use the WDTR register to initialize the watchdog timer.

(6) Setting the watchdog timer start register



Watchdog timer start register (WDTS)

The watchdog timer starts by a WDTS register write instruction.

5. Sample Programs

5.1 Writing to the OFS1 Address

The OFS1 address is located in the flash memory, so write to this address along with the program.
Figure 2 shows how to write a program fragment to the OFS1 address using the extended directive command ".OFSREG" of as30 in C language.

```
#if __WATCH_DOG__ != 0
    _asm(".ofsreg  0FEH");    /* WATCH DOG TIMER START When on Reset */
#else
    _asm(".ofsreg  0FFH");    /* WATCH DOG TIMER STOP When on Reset */
#endif
```

Figure 2. How to Write to the OFS1 Address

5.2 To Use a Watchdog Timer Interrupt

In the example here, the PLL clock (24 MHz) is chosen to be the CPU clock. Also, the CSPROINI and WDTON bits at the OFS1 address are set to 1.
While writing to the WDTR register, increment the display of port P10 every 500 ms. When the output of port P10 becomes 40h, stop writing to the WDTR register and turn off updating of the port 10 display.
When a watchdog timer interrupt is generated, decrement the display of port P10 every 500 ms while writing to the WDTR register in a watchdog timer interrupt handler. When the output of port P10 becomes 00h, turn off updating of the port 10 display.

```
/**********************************************************************************/
/*
/*  M16C/64 Group Program Collection
/*
/*  File name   :   rjj05b1289_int_src.c
/*  CPU         :   M16C/64 Group
/*  Function    :   Operation of Watchdog Timer
/*  Version     :   1.00 (2008-05-21) Initial
/*
/*  Copyright(C)2008, Renesas Technology Corp. , All rights reserved.
/*
/**********************************************************************************/
/* Refer to the corresponding application notes for program specifications.       */

#include    "sfr64.h"

void main(void);
void mcu_init(void);

/**********************************************************************************/
/*      DEFINE
/**********************************************************************************/
void    wdt_int(void);

/**********************************************************************************/
/*      RAM
/**********************************************************************************/

/**********************************************************************************/
/*      ROM
/**********************************************************************************/
```

```
/************************************************************************************/
/*      Pragma
/************************************************************************************/

/************************************************************************************/
/*      Main Program
/************************************************************************************/
void main(void)
{
     unsigned int i;

    mcu_init();                 /* MCU initialize */

    p10 = 0;
    pd10 = 0xff;

    ta0mr = 0x40;               /* Selection of timer mode
                                   Pulse output function select bit (0:Pulse is not output)
                                   Gate function select bit (00:Gate fuction not available)
                                   Count source (01:f8) */
    ta0 = 4000-1;               /* Setting counter value (1msec @24MHz, f8) */

    cpsrf = 0;                  /* Setting clock prescaler reset flag (0:No effect) */

    ta0ic = 0x00;               /* Setting interrupt priority levels in timer A0 */

    ta0s = 1;                   /* TimerA1 count start */

    prcr = 0x02;

    pm12 = 0;                   /* Watchdog Timer Function Select Bit(0 : Watchdog timer interrupt)
                                   */
    prcr = 0x00;

    wdc = 0;                    /* Setting watchdog timer control register
                                   Prescaler select bit is set to 0 (0:Divided be 16) */
    wdtr = 0x00;                /* Setting watchdog timer reset register
                                   Watchdog timer initialize  */
    wdtr = 0xff;                /* Setting watchdog timer value (0x7FFF) */

    i = 0;

    wdts = 1;                   /* Setting watchdog timer start register */

    while (1) {
        while (!ir_ta0ic) {}
        ta0ic = 0x00;
        i++;
        if ( i >= 500 ) {
            i = 0;
            p10++;
        }
        if ( p10 >= 64 ) {
            p10 = 64;
        } else {
            wdtr = 0x00;                /* Setting watchdog timer reset register
                                           Watchdog timer initialize  */
            wdtr = 0xff;                /* Setting watchdog timer value (0x7FFF) */
        }
    }
}

/************************************************************************************/
/*      MCU Initialize
/************************************************************************************/
void mcu_init()
{
     unsigned int i;
    prcr  = 0x03;                /* CM0,CM1,CM2,PLC0,PCLKR register protect off           */
                                 /* PM0,PM1,PM2,TB2SC,INVC0,INVC1 register protect off    */
    pm0   = 0x00;                /* Processor mode: Single-chip mode                      */
    pm1   = 0x08;                /* Watchdog timer function: Watchdog timer interrupt     */
                                 /* Internal reserved area: The entire area usable        */
                                 /* Wait: No wait state                                   */
    cm2   = 0x00;                /* System clock select: Main clock                       */
    cm1   = 0x20;                /* Main clock division select: No division mode          */
    cm0   = 0x08;                /* Main clock division select: CM16 and CM17 enabled     */
    pm20  = 0;                   /* SFR 2waits when PLL on                            */
    plc0  = 0x14;                /* PLL clock Multiply by 4                           */
                                 /* PLL multiplying factor select: Multiply by 8      */
                                 /* Reference frequency counter: Divide by 2          */
    plc07 = 1;                   /* PLL operation enable: PLL ON                      */
    for (i = 0; i < 20000; i++); /* Wait until the PLL clock becomes stable (tsu(PLL)) */
    cm11  = 1;                   /* System clock select: PLL clock                    */
    prcr  = 0x00;                /* Protect on                                        */
}

/******************************************
 * Watchdog timer interrupt routine      *
 ******************************************/
#pragma interrupt wdt_int()
void    wdt_int() {
    unsigned int i = 0;

    wdtr = 0x00;                /* Set the WDTS register in the beginning of the */
    wdtr = 0xff;                /* watchdog timer interrupt routine  */

    while (1) {
        while (!ir_ta0ic) {}
        ta0ic = 0x00;
        i++;

        wdtr = 0x00;
        wdtr = 0xff;

        if(i == 500){
            i = 0;
            if ( p10 != 0 ) {
                p10--;
            }
          }
    }
}
```

5.3 To Reset by a Watchdog Timer Underflow

In the example here, the PLL clock (24 MHz) is chosen to be the CPU clock. Also, the CSPROINI and WDTON bits at the OFS1 address are set to 1.

While writing to the WDTR register, increment the display of port P10 every 500 ms. When the output of port P10 becomes 40h, stop writing to the WDTR register and turn off updating of the port 10 display. The device is reset when the watchdog timer underflows upon reaching the terminal count.

```
/************************************************************************************/
/*
/*  M16C/64 Group Program Collection
/*
/*  File name   :   rjj05b1289_reset_src.c
/*  CPU         :   M16C/64 Group
/*  Function    :   Operation of Watchdog Timer
/*  Version     :   1.00 (2008-05-21) Initial
/*
/*  Copyright(C)2008, Renesas Technology Corp. , All rights reserved.
/*
/************************************************************************************/
/* Refer to the corresponding application notes for program specifications.       */

#include    "sfr64.h"

void main(void);
void mcu_init(void);

/************************************************************************************/
/*      DEFINE
/************************************************************************************/
void    wdt_int(void);

/************************************************************************************/
/*      RAM
/************************************************************************************/
```

```
/**********************************************************************************/
/*      ROM
/**********************************************************************************/

/**********************************************************************************/
/*      Pragma
/**********************************************************************************/

/**********************************************************************************/
/*      Main Program
/**********************************************************************************/
void main(void)
{
     unsigned int i;

    mcu_init();                 /* MCU initialize */

    p10 = 0;
    pd10 = 0xff;

    ta0mr = 0x40;               /* Selection of timer mode
                                   Pulse output function select bit (0:Pulse is not output)
                                   Gate function select bit (00:Gate function not available)
                                   Count source (01:f8) */
    ta0 = 4000-1;               /* Setting counter value (1msec @24MHz, f8) */

    cpsrf = 0;                  /* Setting clock prescaler reset flag (0:No effect) */

    ta0ic = 0x00;               /* Setting interrupt priority levels in timer A0 */

    ta0s = 1;                   /* TimerA1 count start */

    prcr = 0x02;

    pm12 = 1;                   /* Watchdog Timer Function Select Bit(1 : Watchdog timer reset)
                                   */
    prcr = 0x00;

    wdc = 0;                    /* Setting watchdog timer control register
                                   Prescaler select bit is set to 0 (0:Divided be 16) */
    wdtr = 0x00;                /* Setting watchdog timer reset register
                                   Watchdog timer initialize  */
    wdtr = 0xff;                /* Setting watchdog timer value (0x7FFF) */

    i = 0;

    wdts = 1;                   /* Setting watchdog timer start register */

    while (1) {
        while (!ir_ta0ic) {}
        ta0ic = 0x00;
        i++;
        if ( i >= 500 ) {
            i = 0;
            p10++;
        }
        if ( p10 >= 64 ) {
            p10 = 64;
        } else {
            wdtr = 0x00;                /* Setting watchdog timer reset register
                                           Watchdog timer initialize  */
            wdtr = 0xff;                /* Setting watchdog timer value (0x7FFF) */
        }
    }
}

/**********************************************************************************/
/*      MCU Initialize
/**********************************************************************************/
void mcu_init()
{
     unsigned int i;
    prcr  = 0x03;                /* CM0,CM1,CM2,PLC0,PCLKR register protect off          */
                                 /* PM0,PM1,PM2,TB2SC,INVC0,INVC1 register protect off   */
    pm0   = 0x00;                /* Processor mode: Single-chip mode                     */
    pm1   = 0x08;                /* Watchdog timer function: Watchdog timer interrupt    */
                                 /* Internal reserved area: The entire area usable       */
                                 /* Wait: No wait state                                 */
    cm2   = 0x00;                /* System clock select: Main clock                     */
    cm1   = 0x20;                /* Main clock division select: No division mode        */
    cm0   = 0x08;                /* Main clock division select: CM16 and CM17 enabled   */
    pm20  = 0;                   /* SFR 2waits when PLL on                              */
    plc0  = 0x14;                /* PLL clock Multiply by 4                             */
                                 /* PLL multiplying factor select: Multiply by 8        */
                                 /* Reference frequency counter: Divide by 2            */
    plc07 = 1;                   /* PLL operation enable: PLL ON                        */
    for (i = 0; i < 20000; i++); /* Wait until the PLL clock becomes stable (tsu(PLL))   */
    cm11  = 1;                   /* System clock select: PLL clock                      */
    prcr  = 0x00;                /* Protect on                                          */
}

/****************************************
 * Watchdog timer interrupt routine      *
 ****************************************/
#pragma interrupt wdt_int()
void    wdt_int() {
    unsigned int i = 0;

    wdtr = 0x00;                /* Set the WDTS register in the beginning of the */
    wdtr = 0xff;                /* watchdog timer interrupt routine  */

    while (1) {
        while (!ir_ta0ic) {}
        ta0ic = 0x00;
        i++;

        wdtr = 0x00;
        wdtr = 0xff;

        if(i == 500){
            i = 0;
            if ( p10 != 0 ) {
                p10--;
            }
          }
    }
}
```

6. Reference Documents

Hardware manual

M16C/64 Group Hardware Manual
(Get the latest version from the Renesas Technology website.)

Technical updates and technical news

(Get the latest information from the Renesas Technology website.)

Website and Support

Renesas Technology website

http://www.renesas.com/

Inquiries

http://www.renesas.com/inquiry
mailto:csc@renesas.com

REVISION HISTORY

Rev.	Date	Description	
		Page	Summary
1.00	Mar 23,2009	-	First Edition issued

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life
   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.