

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/62

Using the M16C/62 Watchdog Timer

1.0 Abstract

The following article introduces and shows an example of how to set up and use the watchdog timer on the M16C/62 microcontroller (MCU).

2.0 Introduction

The M16C/62 MCU has a built-in watchdog timer, which can be used for a variety of applications. For most applications, it is used to recover MCU processing from a program that is out of control. In some cases, it can be used to preserve processor or firmware status after an application runs out of control.

In this example application, we show you how to set up the watchdog timer, the watchdog interrupt vector, and how the application uses the watchdog timer. This example was written for the MSV1632-Board with an oscillator frequency $X_{in} = 16$ MHz.

3.0 Watchdog Timer Demo

This section discusses what the watchdog timer demo is and how it works. The key components of the program are discussed in the next section; a program listing appears later in the article.

3.1 M16C/62 Watchdog Timer

The M16C/62 watchdog timer is a 15-bit counter using BCLK as the clock source. BCLK and the watchdog prescaler control the length of time before the timer expires. This BCLK-prescaler combination can be used for a wide range of watchdog timing requirements.

A hardware watchdog interrupt is generated after the timer expires and the program executes the watchdog interrupt routine. To prevent the watchdog timer from expiring, the Watchdog Timer Start Register (WDTS) must be written before the timer underflows. For example, if the watchdog timer is set up for 2s, the WDTS register must be written to within 2s so that the timer does not expire.

For this demo, the timer was set up for 2.097s.

3.2 Watchdog Interrupt Routine

After the watchdog timer expires, a hardware interrupt is generated. An interrupt service routine must be in place for the program to execute when this interrupt occurs. This interrupt routine can be used to store program parameters or register status in RAM. As an added fail-safe feature for your application, it may be used to reset the M16C/62 MCU (the M16C/62 has a built-in software reset).

In this demo, the 3 LEDs are turned on and the program loops inside the watchdog service routine. The only way to exit from the routine is by pressing the reset switch, SW5.

3.3 The Demo Application

The demo application uses two timers (Timer A0, A1), the AD converter using AN0, and I/O ports. Timer A0's output is used as the clock source of Timer A1. Timer A1 is preloaded with the ADC value of the R24 potentiometer and is then used to set up how fast the LED's LED2–4 blink and the WDTs register is written to. The I/O ports are used to turn on or off the LED's LED2–4 and to read the status of the switches SW1–SW4. By adjusting R24 from full clockwise (CW) position to full counterclockwise position (CCW), the period of when WDTs is written varies also. The LEDs will be blinking fast at full CW and very slow (about 2.5s interval) at full CCW. At full CCW, the time period of when WDTs is written is greater than 2.097s, which will then trigger a watchdog interrupt. However, still at full CCW, if any of the switches is pressed within 2s, the watchdog timer is restarted and thus, a watchdog timer interrupt is not generated.

4.0 Watchdog Timer Setup

A watchdog timer interrupts after a certain time has expired. As mentioned earlier, the M16C/62 watchdog timer can be set up for various time periods by configuring the BCLK and watchdog prescaler. The timer period can be calculated from the following equation. These parameters will be discussed later in this section. For more detailed information, see the M16C/62 datasheet.

$$\text{Watchdog Timer Period} = (\text{Prescaler} \times \text{Timer Count}) / \text{BCLK}$$

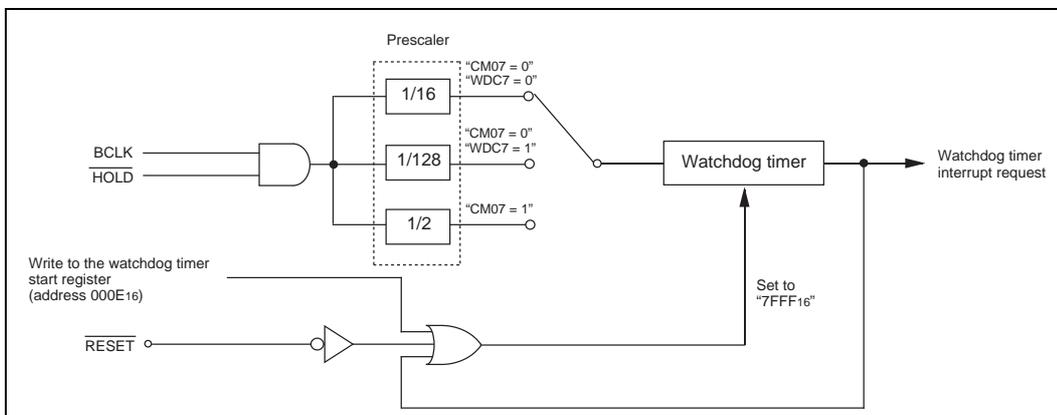


Figure 1 Block Diagram of Watchdog Timer

4.1 BCLK

The clock source of the timer is BCLK, which is the CPU clock for the M16C/62. The value of BCLK can be modified by changing the oscillator circuits of the device or by changing setting in the clock control registers (see “Clock Control” from the datasheet). BCLK can use Xin (f1), XCin (fc), or clock divider output (f2, f4, f8, f16, f32). Modifying the BCLK will then modify the frequency the timer counts down and program operating speed. For this demo, the clock divider output f8 was used as the BCLK. With an Xin frequency of 16 MHz, BCLK frequency is 2 MHz.

4.2 Prescaler

Besides BCLK, the other parameter that can adjust the timer is the watchdog prescaler. The prescaler further divides BCLK for larger time periods. The prescaler that can be used differs depending on whether Xin or XCin is used as the BCLK source. If Xin is used, the prescaler can be either div 16 or div 128. If XCin is used, the prescaler is fixed to div 2.

For this demo, the prescaler used is div 128 since our BCLK source is Xin.

4.3 Timer Count

Besides BCLK and the prescaler, the other parameter is the timer count. This parameter, however, cannot be modified. Regardless of what value is written to the WDTS register, the default value of 07FFFh is loaded into the timer.

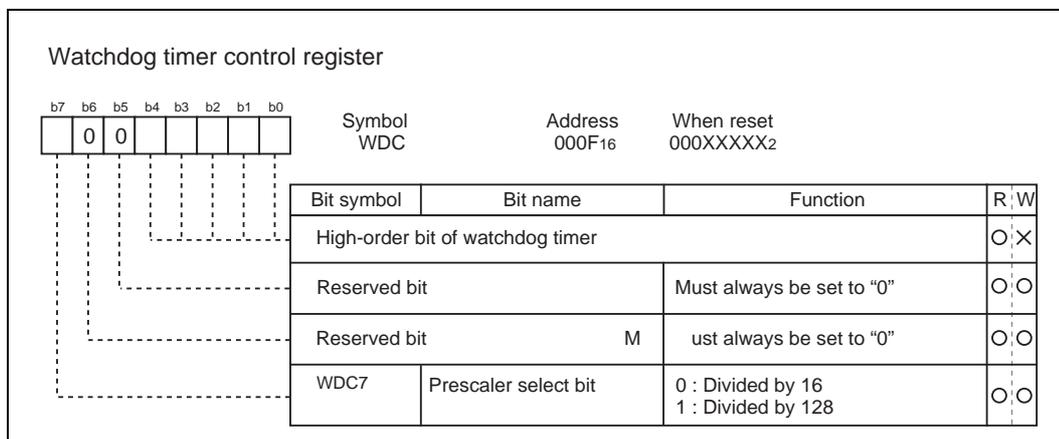


Figure 2 Watchdog Timer Control and Start Registers

5.0 Reference

Renesas Technology Corporation Semiconductor Home Page

<http://www.renesas.com>

E-mail Support

support_apl@renesas.com

Data Sheet

- M16C/62 datasheets, 62aeds.pdf

User's Manual

- M16C/62 User's Manual, 62eum.pdf
- M16C/60 and M16C/20 C Language Programming Manual, 6020EC.pdf
- Application Note: Writing Interrupt Handlers in C for the M16C
- NC30 Ver. 4.0 User's Manual, NC30UE.PDF

6.0 Software Code

The example program was written to run on the MSV1632-62 Board but could be modified to implement in a user application. The program is written in C (Renesas' NC30 Compiler).

```

/*****
*
*   File Name: main_wdt.c
*
*   Content: This program demonstrates how to setup and use the watchdog
*            function of an MCU when the firmware program malfunctions
*
*
*           The watchdog is setup to generate an interrupt after 2.097s.
*           To prevent the watchdog interrupt from occurring, either of the
*           following conditions listed below must be true:
*           1. at least one of the switches (SW1 - SW4) is pressed every 2s; or,
*           2. R24 should NOT be in full CCW position. The position of R24 is
*              read by the ADC and used to vary Timer 1 period. R24 full CW to
*              full CCW corresponds to 8.23ms to 2.0986s. At a value of
*              greater than 254, the WDI will occur.
*
*
*           This program was written to run on the MSV1632-62 SKP Board.
*
*           Copyright 2003 Renesas Technology Corporation, Inc.
*           All rights reserved
*
*=====
*           $Log:$
*=====*/

#include "sfr62.h"
#pragma INTERRUPT /B TimerA1_ISR

#define      sw1      p8_2
#define      sw2      p8_3
#define      sw3      p8_4
#define      sw4      p9_7

#define      red_led      p7_2
#define      yellow_led p7_4
#define      green_led p8_0

```

```

void WD_Init (void);           //routine that initializes watchdog operations
void WD_Loop_ISR(void);       //routine when a watchdog interrupt is
                               // generated (timer expires)

static int value;
static unsigned int out1;

/*****
Name:    main
Parameters:
Returns:
Description:  Main program loop and initialization
*****/
main() {

    WD_Init();                // initialize watchdog block

    pd7_2 = 0x1;              // Change port 7 to all outputs (connected to LEDs)
    pd7_4 = 0x1;
    pd8_0 = 0x1;

    red_led = 0x1;           // turn on LEDs
    green_led = 0x1;
    yellow_led = 0x1;

    // ***** USE A/D FOR READING POT. VALUE *****
    // Set up A/D register for AN0
    // - AN0 selected, One shot mode, Software trigger, Frequency /4

    adcon0 = 0x00;
    adcon1 = 0x20;           // Set up 8 bit conversion & Vref connected
    adcon2 = 0x01;           // Set up sample and hold

    // ***** USE TIMERS FOR SETTING BLINK RATE *****
    // Timer A0 mode register
    // - Timer mode, no pulse output,not using Gate function,count source is
    // f32 (500KHz)

    ta0mr = 0x80;
    ta0ud = 0;               // Set timer for down count
    ta0 = 0x1013;           // Preload Timer A0 to overflow every
                               // 10ms (ta0 = 10ms*500KHz)
                               // 10ms x 255 (2.55s) > 2.097s watchdog setting

    // Timer A1 mode register
    // -event counter mode (Timer A0 overflow, count on falling edge)

```

```

talmr = 0x01;
taltgh = 1;
taltgl = 0;
talud = 0;           // Set timer for down count
tal = 0;            // Preload Timer A1

talic = 0x05;       // TA1 Interrupt enabled, Level 5

ta0s = 1;           // Start the timers
tals = 1;

asm("FSET I");      // Turn on interrupts

// ***** PROGRAM LOOP *****
while(1) {

    adst = 1;        // Start A2D conversion
    while(adst == 1); // wait for A/D conversion start bit to return
                        // to 0
    value = ad0;     // read value from A/D register and preload
                    // TimerA1,
    tal = value;     // this value is used to vary the blink rate

    if(sw1 == 0 || sw2 == 0 || sw3 == 0 || sw4 == 0) { // check if any of the
                                                        // switches is pressed
        green_led = 1;
        wdts = 0;           // restart watchdog timer
    }
}
}
/*****
Name:    WD_Init
Parameters:
Returns:
Description:  Initializes variables needed for watchdog operations.
              This demo is written for the MSV1632 Board running at Xin =
              16MHz. BCLK, which is the clock source of the watchdog
              block, is configured to run at 2MHz.

              With the setup shown here, the watchdog register must be
              written to within 2 seconds or a watchdog interrupt will occur.

*****/

```

```
void WD_Init(void){
    cm06 = 1;           // BCLK = 2MHz (Xin div by 8, default)

    wdc7 = 1;          // prescaler is div by 128
                       // Watchdog Timer Period = 128 x 32768 / 2MHz
                       // = 2.097s
    wdts = 0;          // start watchdog timer by writing any value to
                       // wdts register (value always reset to 0x7fff when
                       // written to)
}
```

Name: WD_Loop_ISR

Parameters:

Returns:

Description: When a watchdog interrupt occurs, the program gets stuck here so we know the watchdog timer expired. All LED's will be ON.

For some applications, you can use this ISR to save some values in RAM or other processes for debug purposes.

```
void WD_Loop_ISR (void){
```

```
    while (1){
        red_led = 1;           // turn on all LEDs
        yellow_led = 1;
        green_led = 1;
    }
}
```

Name: TimerA1_ISR

Parameters:

Returns:

Description: TIMER INTERRUPT ROUTINE - This routine turns on one LED at a time and shifts that LED left to right (D3 to D6). The LEDs are connected to the upper byte of P7 and we don't want to change the data on the lower byte.

If none of the switches are pressed, the watchdog timer is restarted every time a Timer 1 interrupt occurs. If the period of Timer 1 exceeds 2.097s (R24 potentiometer in almost full CCW), watchdog interrupt will occur.

```

void TimerA1_ISR( void) {

    static unsigned int out1;

    wdts = 0;                                // restart Watchdog Timer

    ++out1;
    if (out1 > 3)
        out1 = 0;
    switch (out1){
        case 1:
            red_led = 1;
            yellow_led = 0;
            green_led = 0;
            break;
        case 2:
            red_led = 0;
            yellow_led = 1;
            green_led = 0;
            break;
        case 3:
            red_led = 0;
            yellow_led = 0;
            green_led = 1;
            break;
        default:
            red_led = 0;
            yellow_led = 0;
            green_led = 0;
    }
}
*****

```

In order for this program to run properly, the Watchdog Timer and TimerA1 interrupt vector needs to point to the service routines for those interrupts. The interrupt vector table information is included in the startup file "sect30.inc". Insert the function label "TimerA1_ISR" and the function label "WD_Loop_ISR WD_Loop_ISR" into the interrupt vector table locations as shown below.

```

;*****
;
; sect30.inc : Customized section and macro definitions for the M30624
;             (M16C/62) microcontroller using the NC30 compiler.
;
; Description : This file is specific to the M30624 microcontroller and adapted
;              for use with the MSV1632 Starter Kit. UART1 interrupt
;              vectors are used for the Starter Kit debugger.
;
;
;

```

```

;          Copyright 2003 Renesas Technology Corporation, Inc.
;          All Rights Reserved.
;
;
;          $Id:
;
;*****
;-----
; variable vector section
; For proper interrupt operation, replace "dummy_int" with the assembler
; label or absolute address of the interrupt service routine
;-----
        .section      vector          ; variable vector table
        .org      VECTOR_ADR

        .lword dummy_int             ; BRK (vector 0)
        .org      (VECTOR_ADR+16)
        .lword dummy_int             ; int3(for user) (vector 4)
        .lword dummy_int             ; timerB5(for user) (vector 5)
        .lword dummy_int             ; timerB4(for user) (vector 6)
        .lword dummy_int             ; timerB3(for user) (vector 7)
        .lword dummy_int             ; si/o4 /int5(for user) (vector 8)
        .lword dummy_int             ; si/o3 /int4(for user) (vector 9)
        .lword dummy_int             ; Bus collision detection(for user) (v10)
        .lword dummy_int             ; DMA0 (for user) (vector 11)
        .lword dummy_int             ; DMA1 (for user) (vector 12)
        .lword dummy_int             ; Key input interrupt(for user) (vect 14)
        .lword dummy_int             ; A-D(for user) (vector 14)
        .lword dummy_int             ; uart2 transmit(for user) (vector 15)
        .lword dummy_int             ; uart2 receive(for user) (vector 16)
        .lword dummy_int             ; uart0 transmit(for user) (vector 17)
        .lword dummy_int             ; uart0 receive(for user) (vector 18)
        .lword 0ff900h                ; uart1 transmit-used by ROM Monitor(vector 19)
        .lword 0ff900h                ; uart1 receive-used by ROM Monitor(vector 20)

        .lword dummy_int             ; timer A0(for user) (vector 21)
        .glob      _TimerA1_ISR;
.lword _TimerA1_ISR;                 ; timer A1(for user) (vector 22)
        .lword dummy_int             ; timer A2(for user) (vector 23)
        .lword dummy_int             ; timer A3(for user) (vector 24)
        .lword dummy_int             ; timer A4(for user) (vector 25)

        .lword dummy_int             ; timer B0(for user) (vector 26)
        .lword dummy_int             ; timer B1(for user) (vector 27)
        .lword dummy_int             ; timer B2(for user) (vector 28)
        .lword dummy_int             ; int0 (for user) (vector 29)
        .lword dummy_int             ; int1 (for user) (vector 30)
        .lword dummy_int             ; int2 (for user) (vector 31)

        .lword dummy_int             ; vector 32 (for user or MR30)
        .lword dummy_int             ; vector 33 (for user or MR30)
        .lword dummy_int             ; vector 34 (for user or MR30)

```

```

        .lword dummy_int          ; vector 35 (for user or MR30)
        .lword dummy_int          ; vector 36 (for user or MR30)
        .lword dummy_int          ; vector 37 (for user or MR30)
        .lword dummy_int          ; vector 38 (for user or MR30)
        .lword dummy_int          ; vector 39 (for user or MR30)
        .lword dummy_int          ; vector 40 (for user or MR30)
        .lword dummy_int          ; vector 41 (for user or MR30)
        .lword dummy_int          ; vector 42 (for user or MR30)
        .lword dummy_int          ; vector 43 (for user or MR30)
        .lword dummy_int          ; vector 44 (for user or MR30)
        .lword dummy_int          ; vector 45 (for user or MR30)
        .lword dummy_int          ; vector 46 (for user or MR30)
        .lword dummy_int          ; vector 47 (for user or MR30)
        ;
;=====
; fixed vector section
;-----
        .section      fvector          ; fixed vector table
;=====
; special page definition
;-----
; Set-up special page vector table. Calls the macro "SPECIAL" to put
; the jump addresses of functions defined as special page into the
; special page vector table. Uncomment the line below that corresponds
; to the C function defined using the "#pragma SPECIAL" directive. See
; the M16C Software Manual and the NC30 manual for more information
; on special page vectors.
;-----
;     SPECIAL 255 ; example use
;     SPECIAL 254
;     SPECIAL 253
;     :
;     etc
;     SPECIAL 24
;     SPECIAL 23
;     SPECIAL 22
;     SPECIAL 21
;     SPECIAL 20
;     SPECIAL 19
;     SPECIAL 18
;
;

```

```

;=====
; fixed vector section. The 7 or'ed values below (commented out) are for
; specifying the ID codes for serial I/O flash programming
; (highest 8 bits of the vectors). See data sheets for
; more information. Current setting = all zeros by default.
; The highest 8 bits of the reset vector is the parallel protection
; 'register'. Caution! Setting these codes could result in loss of
; all flash programming. See M30624 data sheets before operating
; on these values.
;-----
                .org    0fffdch
UDI:
                .lword  dummy_int ; | 0ff000000h
OVER_FLOW:
                .lword  dummy_int ; | 0ff000000h
BRKI:
                .lword  dummy_int
ADDRESS_MATCH:
                .lword  dummy_int ; | 0ff000000h
SINGLE_STEP:
                .lword  dummy_int ; | 0ff000000h
WDT:
                .glb          _WD_Loop_ISR WD_Loop_ISR;
                .lword  _WD_Loop_ISR WD_Loop_ISR dummy_int ;
DBC:
                .lword  dummy_int ; | 0ff000000h
NMI:
                .lword  dummy_int ; | 0ff000000h
RESET:
                .lword  start ; | 0ff000000h
;

```

Keep safety first in your circuit designs!

- Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
- Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss arising from these inaccuracies or errors.
Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (<http://www.renesas.com>).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.