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.

RENESAS

7542 Group

Sample Program Serial I/O 1(UART)-LED Pattern Control

Introduction

This sample program uses the following functions:

- Timer X (Timer mode): The period of the main loop function.
- INT0 (Falling edge active): Transmit trigger.
- Serial I/O 1 (UART mode): Full duplex communication between two MCUs.
- Output Port (P3$_0$-P3$_2$, P0$_3$): LED0, LED1, LED2, LED3 control.

Target Device

The explanation of this issue is applied to the following condition:

- Target MCU: 7542 Group
- Oscillation frequency: 8MHz
- Memory size: ROM 32KB, RAM 1KB

Contents

1. Operation

The program is explained in following.

- Serial I/O 1(UART) is used in this sample program, communication format is shown below:
    — Transfer Baud Rate: 9615bps;
    — Transfer Data Length: 8 bits;
    — Stop Bit: 1 bit;
    — Parity Bit: Even Parity.

One byte is transferred from the transmitter to the receiver when SW1 is pressed. Initial value of transmitting data is 00001111B. It decreases by one when SW1 pressed each time until the data equals 0, then initial value is reloaded.

Circuit diagram is shown in the figure 1.
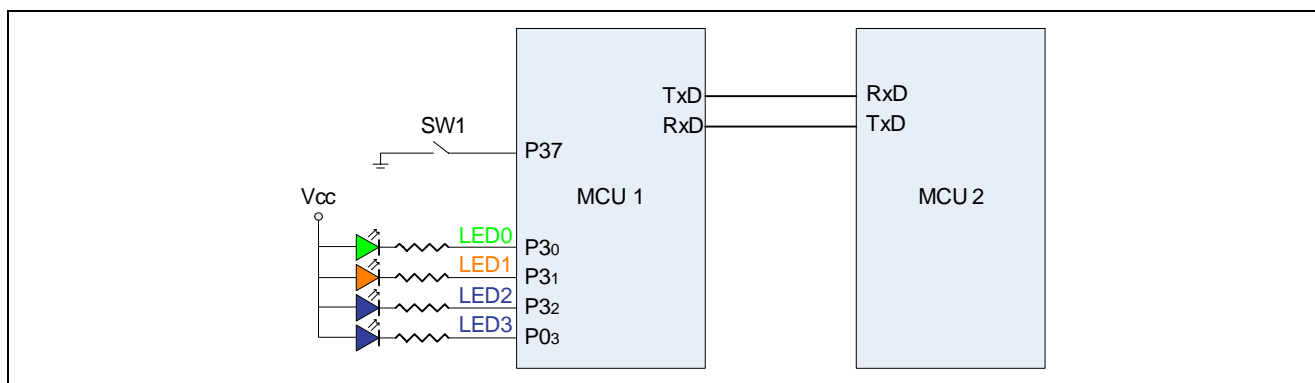
Figure 1   Circuit Diagram

The receiver turns on corresponding LED according to received data.

Received data	LED3	LED2	LED1	LED0
0x0f	●	●	●	●
0x0e	●	●	●	○
0x0d	●	●	○	●
0x0c	●	●	○	○
0x0b	●	○	●	●
0x0a	●	○	●	○
0x09	●	○	○	●
0x08	●	○	○	○
0x07	○	●	●	●
0x06	○	●	●	○
0x05	○	●	○	●
0x04	○	●	○	○
0x03	○	○	●	●
0x02	○	○	●	○
0x01	○	○	○	●
0x00	○	○	○	○

● ● ● ON
○ OFF

Figure 2   Received Data - LED Display Pattern

2. Contents

2.1 Global Variables

Table 1. Global variable

Name	Initial Value	Size (Byte)	Description	Min (H)	Max (H)	Flag
g_mode	00H	1	Mode flag: 0 Receive; 1 Transmit	00	01	
g_error	00H	1	Error flag	00	FF	
g_tr_data	0FH	1	Transmit data buffer	00	0F	
g_re_data	00H	1	Receive data buffer	00	FF	
g_key_state	00H	1	Key state flag	00	04	

2.2 Timer X (Timer mode)

Timer X is used in timer mode. The standard period of 20ms is used for main cycle. Interrupt of Timer X isn't being used.

$$8MHz \times f16 \times 250 \times 40 = 20ms \quad \text{Main cycle 20ms}$$

Setting timer count source register

b7   b0

| | | | | | | 0 | 0 | TCSS [Address $002A_{16}$]

Timer x count source select bits = 00: 16 division is selected

Timer A count source select bit

Timer B count source select bit
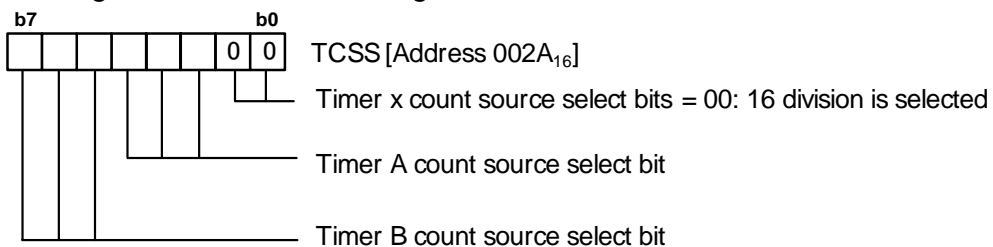
Setting Timer X prescaler register

b7   b0

$F9_{16}$ PREX [Address $002C_{16}$]

Setting Timer X count register

b7   b0

27_{16} TX [Address $002D_{16}$]

Setting Timer X mode register
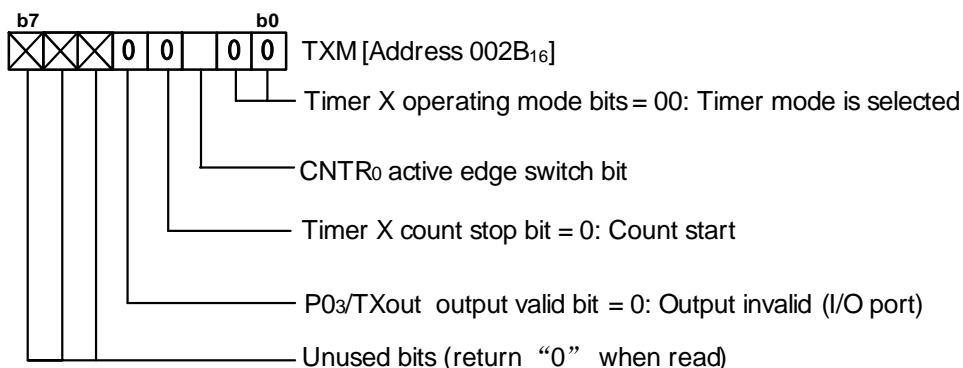
Figure 3   Setting of Timer X

2.3   INT0 (falling edge active)

Interrupt of INT0 is used. The setup of INT0 is shown in the following.

Setting interrupt edge selection register
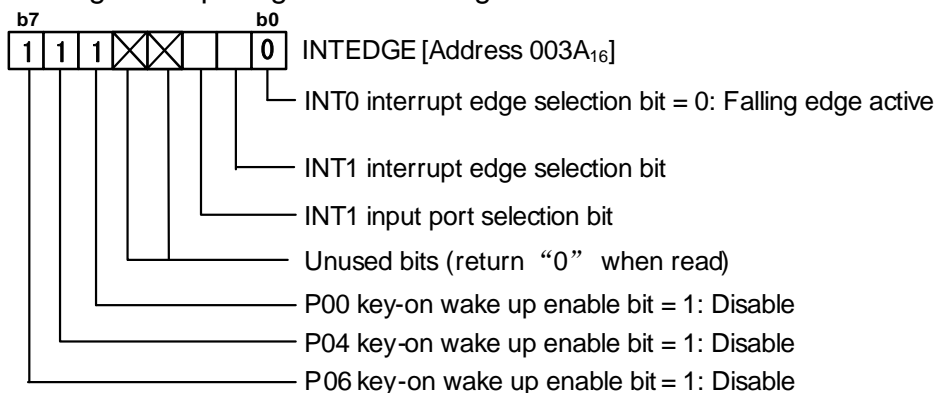
Setting interrupt control register 1
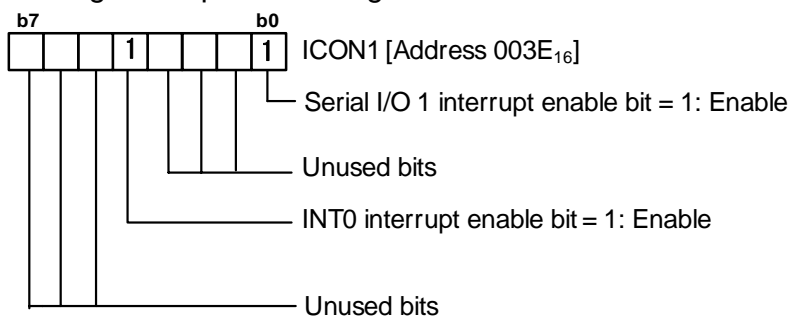
Figure 4   Setting of INT0

2.4   Serial I/O 1 (UART mode)

For clock asynchronous serial I/O (UART), the baud rate and transfer formats used by a transmitter and receiver must be identical. When a byte is received, an interrupt will happen.

The setup of Serial I/O 1 is shown in the following.

Setting UART 1 control register

b7   b0

| ✕ | ✕ | ✕ | 0 | 0 | 0 | 1 | 0 |

UART1CON [Address $001B_{16}$]

Character length selection bit = 0: 8 bits

Parity enable bit = 1: Parity checking enable

Parity selection bit = 0: Even parity

Stop bit length selection bit = 0: 1 stop bit

$P1_1/TxD_1$ P-channel output disable bit = 0: CMOS output

Unused bits (return "0" when read)
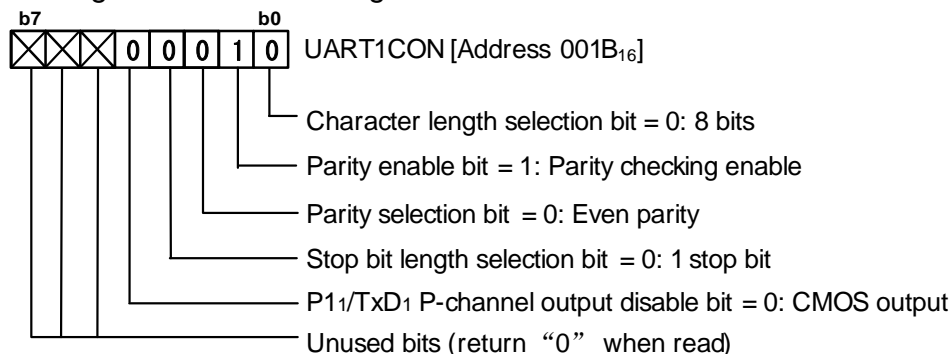
Setting Baud rate generator register

b7   b0

| 33_{16} |

BRG1 [Address $001C_{16}$]

Setting Serial I/O 1 control register

b7   b0

| 1 | 0 | 1 | 1 | | | 0 | 0 |

SIO1CON [Address $001A_{16}$]

BRG count source selection bit = 0: f(Xin)

Serial I/O 1 synchronous clock selection bit = 0: BRG output divided by 16

$\overline{SRDY1}$ output enable bit

Transmit interrupt source selection bit

Transmit enable bit = 1: Enable

Receive enable bit = 1: Enable

Serial I/O 1 mode selection bit = 0: UART mode

Serial I/O 1 enable bit = 1: Enable
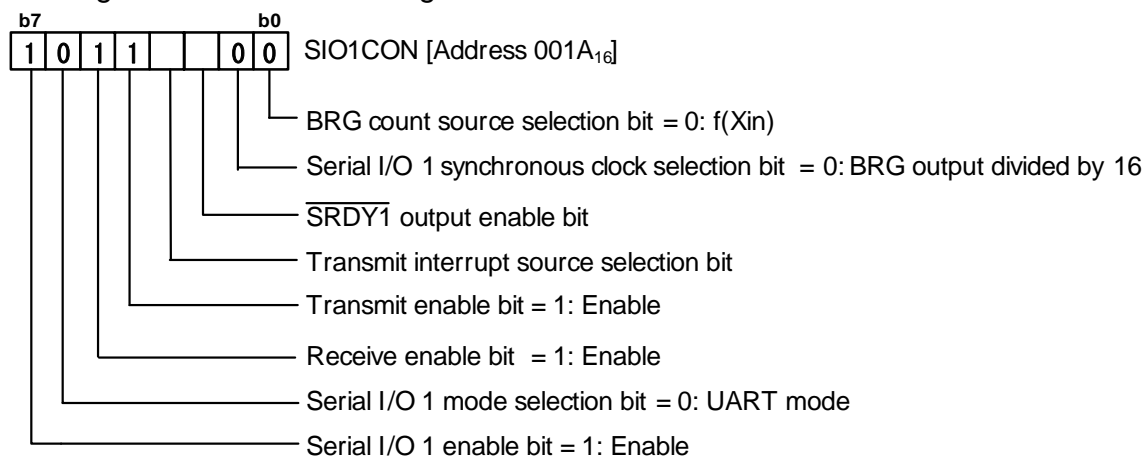
Figure 5   Setting of Serial I/O 1

2.5   I/O Ports

LED0, LED1, LED2, LED3 are controlled by $P3_0$, $P3_1$, $P3_2$, $P0_3$.

Output of the ports are refreshed every 20ms.

The setup of the ports is shown in the following.

Setting Port P0P3 drive capacity control register

b7							b0

DCCR [Address 0015_{16}]

— Unused bits

— P3$_{0-2}$, P0$_3$ drive capacity

— Unused bits
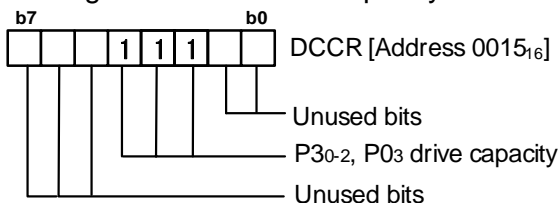
Setting Port P0 register

b7							b0

P0 [Address 0000_{16}]

— Unused bits

— P0$_3$ is set to "1"

— Unused bits
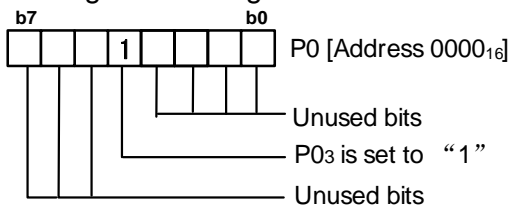
Setting Port P0 direction register

b7							b0

P0D [Address 0001_{16}]

— Unused bits

— P0$_3$ is set to output

— Unused bits
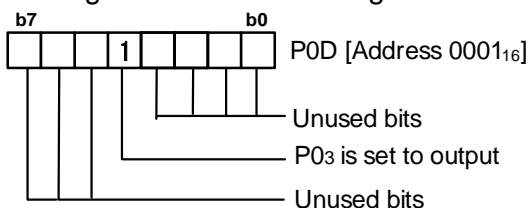
Setting Port P3 register

b7							b0

P3 [Address 0006_{16}]

— P3$_{0-2}$ is set to "1"

— Unused bits
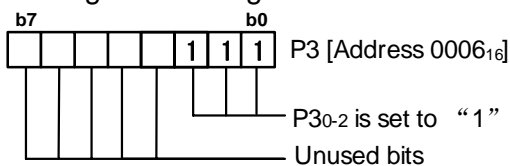
Setting Port P3 direction register

b7							b0

P3D [Address 0007_{16}]

— P3$_{0-2}$ is set to output

— Unused bits

— P3$_7$ is set to input
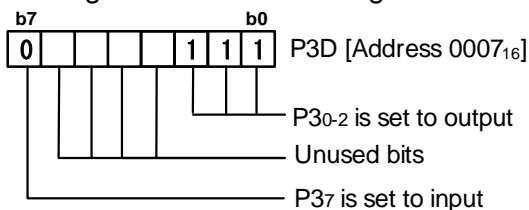
Figure 6   Setting of I/O Ports

3. Flow Chart

3.1 Initialize & Main Loop

main

__asm("SEI") : Interrupt Disable

PREX = 50 - 1 : Prescale = 50 - 1

TX = 25 - 1 : Timer X = 25 - 1

IREQ2.3 = 0 : Clear Timer X interrupt request bit

SFR initialize
sfr_init

IREQ2.3 = 0? : 10ms Delay for external OSCI stabilization
Yes / No

IREQ2.3 = 0 : Clear Timer X interrupt request bit

CPUM = 0x00 : Change to main clock

PREX = 250 - 1 : Prescale = 250 - 1

TX = 40 - 1 : Timer X = 40 - 1

__asm("CLI") : Interrupt enable

IREQ2.3 = 0? : 20ms main cycle
Yes / No

IREQ2.3 = 0

g_error != 0? : Check error flag
No / Yes

Error processing
error_process

Check keys
key_check

Mode processing
mode_func

Figure 7   Initialize & Main Loop Flow Chat

3.2   SFR Initialize


```
                    ┌──────────────┐
                    │   Sfr_init   │
                    └──────────────┘
                            │
                    ┌──────────────┐
                    │ DCCR = 0x1c  │ : P0 P3 drive capacity configuration
                    └──────────────┘
                            │
                    ┌──────────────┐
                    │ P0  = 0x08   │ : P03 = H(LED3)
                    └──────────────┘
                            │
                    ┌──────────────┐
                    │ P0D = 0xff   │ : PD03 = Output(LED3),
                    └──────────────┘   unused pins set to output
                            │
                    ┌──────────────┐
                    │ P1D = 0x1f   │ : Unused pins output
                    └──────────────┘
                            │
                    ┌──────────────┐
                    │ P2D = 0xfe   │ : PD20 = Input(AD0),
                    └──────────────┘   unused pins set to output
                            │
                    ┌──────────────┐
                    │ P3  = 0x07   │ : P30-2 = H(LED0,1,2)
                    └──────────────┘
                            │
                    ┌──────────────┐
                    │ P3D = 0x6f   │ : P30-2 = Output(LED0,1,2),
                    └──────────────┘   P37 = Input (SW1)
                            │
                    ┌────────────────┐
                    │ INTEDGE = 0xe0 │ : P04, P06 Key-on Wakeup Disable,
                    └────────────────┘   INT0 Falling Edge Active
                            │
                    ┌────────────────┐
                    │ UART1CON= 0x02 │ : Parity checking enable
                    └────────────────┘
                            │
                    ┌──────────────┐
                    │ BRG1 = 52 - 1│ : 8MHz * 1/16 * 1/52 = 9615 bps
                    └──────────────┘
                            │
                    ┌────────────────┐
                    │ SIO1CON = 0xb0 │ : Serial I/O 1 enable; Transmit enable; Receive enable
                    └────────────────┘
                            │
                    ┌──────────────┐
                    │ TABM  = 0x0a │ : Stop Timer A and Timer B
                    └──────────────┘
                            │
                    ┌──────────────┐
                    │ IREQ1 = 0    │ : Clearing interrupt request bit
                    └──────────────┘
                            │
                    ┌──────────────┐
                    │ IREQ2 = 0    │ : Clearing interrupt request bit
                    └──────────────┘
                            │
                    ┌──────────────┐
                    │ ICON1 = 0x11 │ : INT0 interrupt enable,
                    └──────────────┘   Serial I/O1 receive interrupt enable
                            │
                    ┌──────────────┐
                    │     RTS      │
                    └──────────────┘
```

Figure 8   SFR Initialize Flow Chat

3.3 Mode Processing
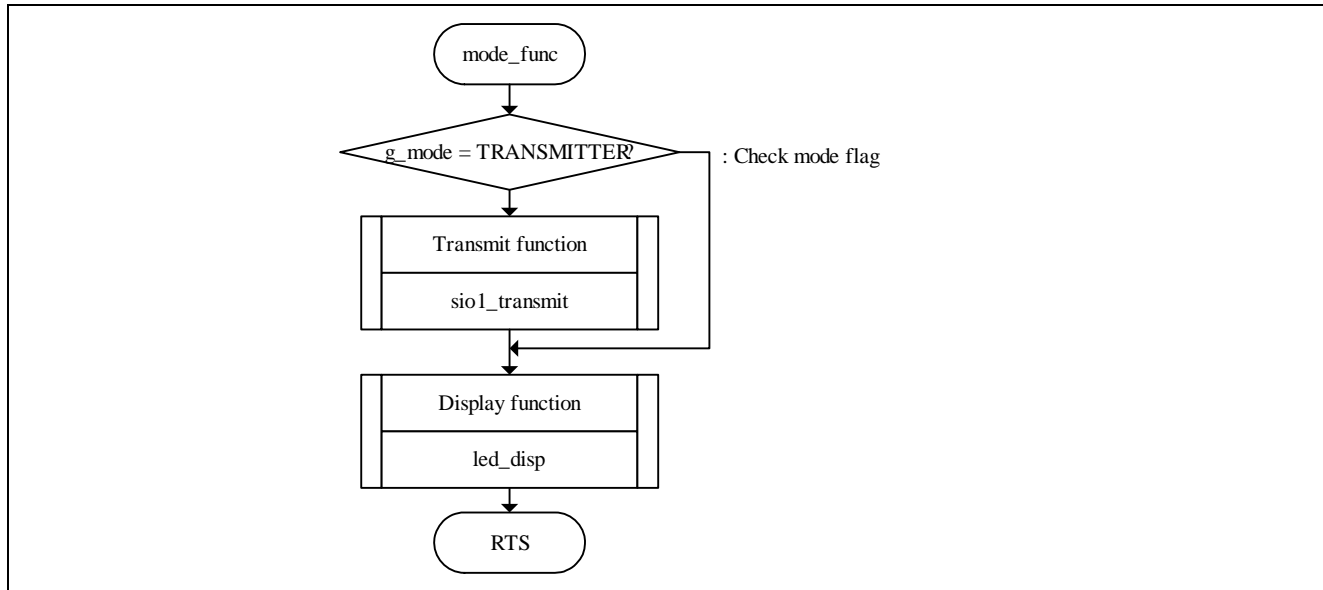
Figure 9   Mode Processing Flow Chat

3.4 Output Ports (P3~0-2~, P0~3~): LED Control

```
                        ┌──────────────┐
                        │   led_disp   │
                        └──────┬───────┘
                               │
                          ◇─────────◇      No
                       g_re_data bit 0=1? ────────┐
                          ◇─────────◇             │
                               │ Yes              │
                        ┌──────────────┐          │
                        │   LED0 = 0   │  :Turn on LED0
                        └──────┬───────┘          │
                               │◄─────────────────┘
                        ┌──────────────┐
                        │   LED0 = 1   │  :Turn off LED0
                        └──────┬───────┘
                               │
                          ◇─────────◇      No
                       g_re_data bit 1=1? ────────┐
                          ◇─────────◇             │
                               │ Yes              │
                        ┌──────────────┐          │
                        │   LED1 = 0   │  :Turn on LED1
                        └──────┬───────┘          │
                               │◄─────────────────┘
                        ┌──────────────┐
                        │   LED1 = 1   │  :Turn off LED1
                        └──────┬───────┘
                               │
                          ◇─────────◇      No
                       g_re_data bit 2=1? ────────┐
                          ◇─────────◇             │
                               │ Yes              │
                        ┌──────────────┐          │
                        │   LED2 = 0   │  :Turn on LED2
                        └──────┬───────┘          │
                               │◄─────────────────┘
                        ┌──────────────┐
                        │   LED2 = 1   │  :Turn off LED2
                        └──────┬───────┘
                               │
                          ◇─────────◇      No
                       g_re_data bit 3=1? ────────┐
                          ◇─────────◇             │
                               │ Yes              │
                        ┌──────────────┐          │
                        │   LED3 = 0   │  :Turn on LED3
                        └──────┬───────┘          │
                               │◄─────────────────┘
                        ┌──────────────┐
                        │   LED3 = 1   │  :Turn off LED3
                        └──────┬───────┘
                               │
                        ┌──────────────┐
                        │     RTS      │
                        └──────────────┘
```

Figure 10   LED Control Flow Chat
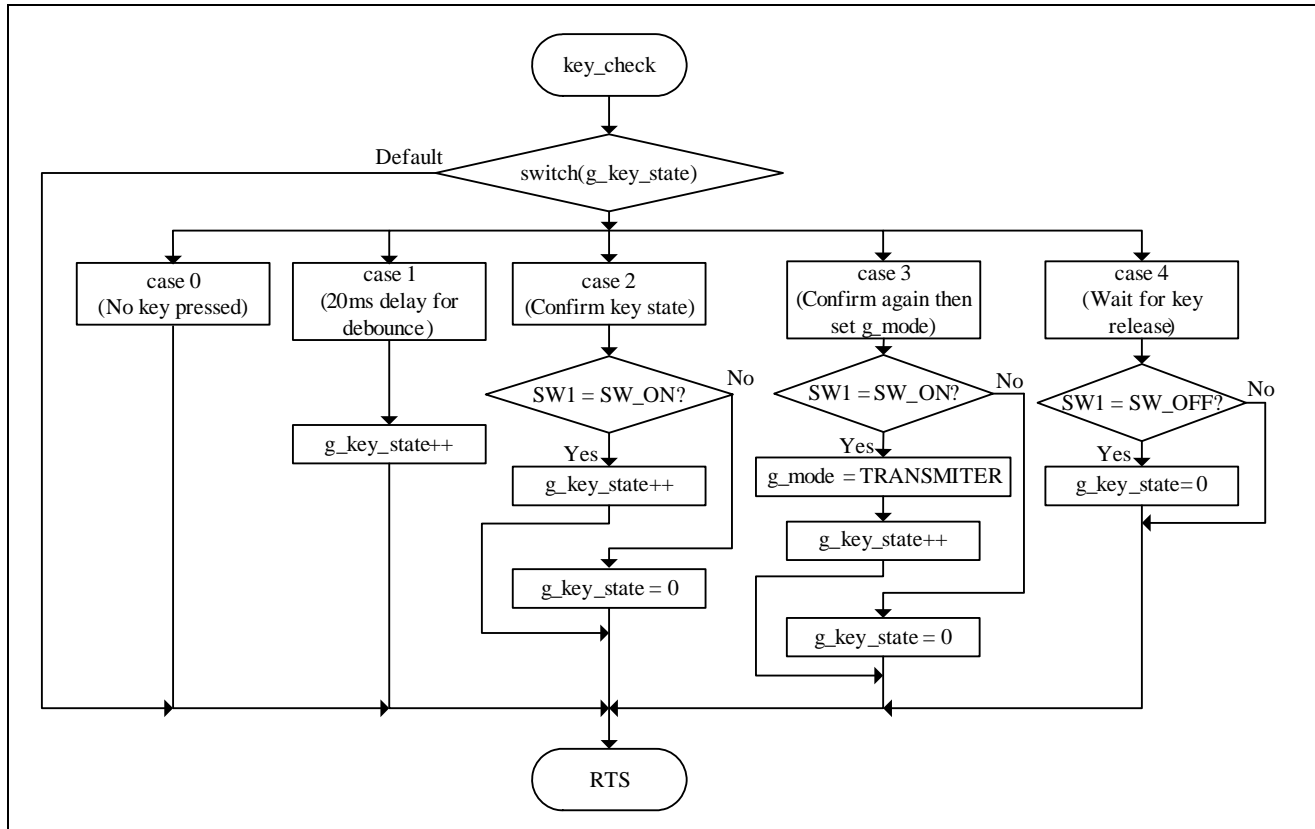
3.5 Key Check

Figure 11   Key Check Flow Chat

3.6 Serial I/O 1 Transmit
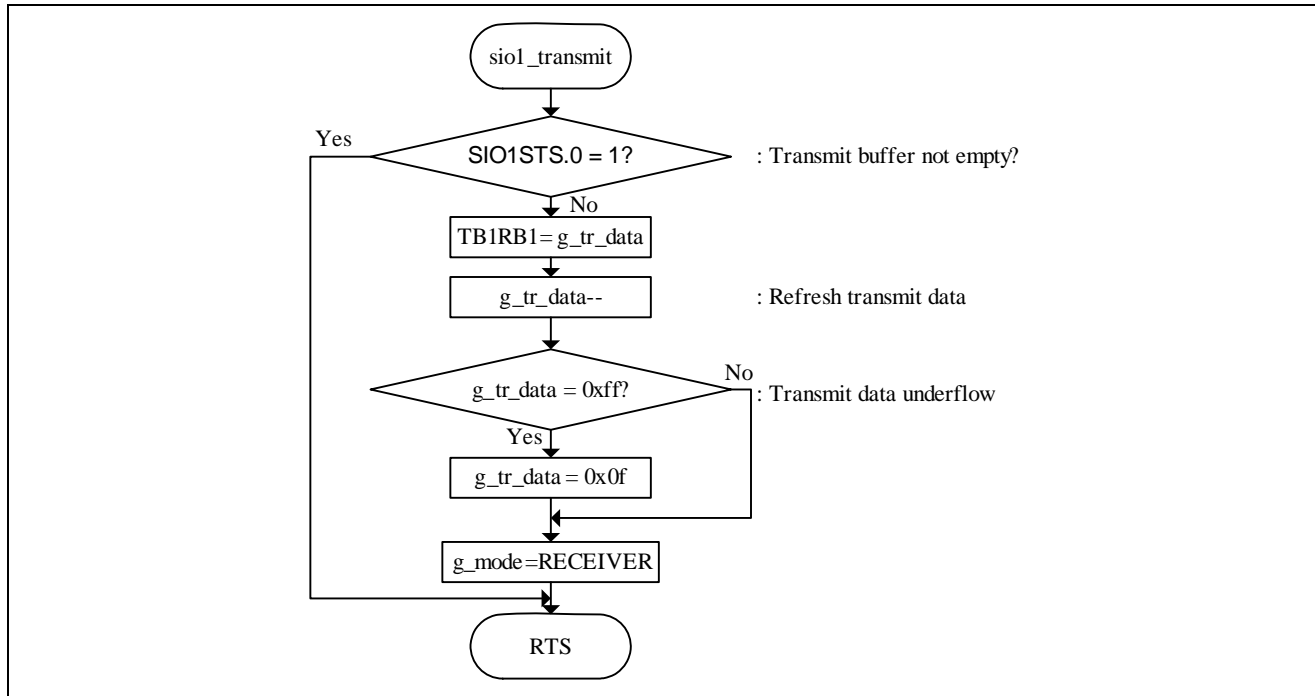
Figure 12   Serial I/O 1 Transmit Flow Chat

3.7 INT0 & Serial I/O 1 Interrupt Handler
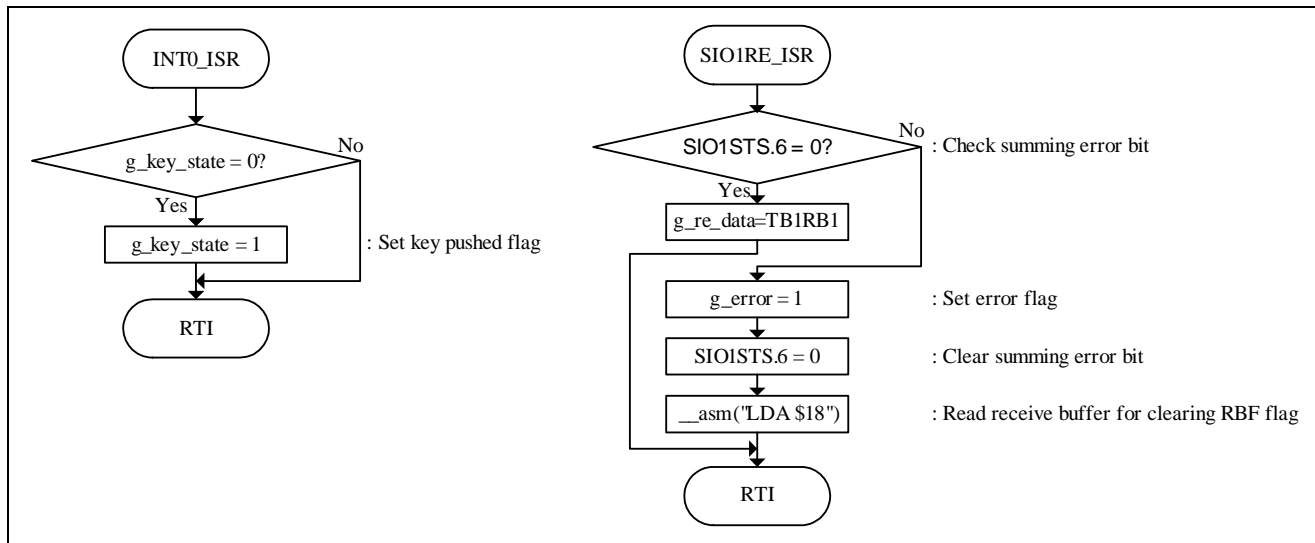
Figure 13   INT0 & Serial I/O 1 Interrupt Handler Flow Chat

3.8   Error Processing
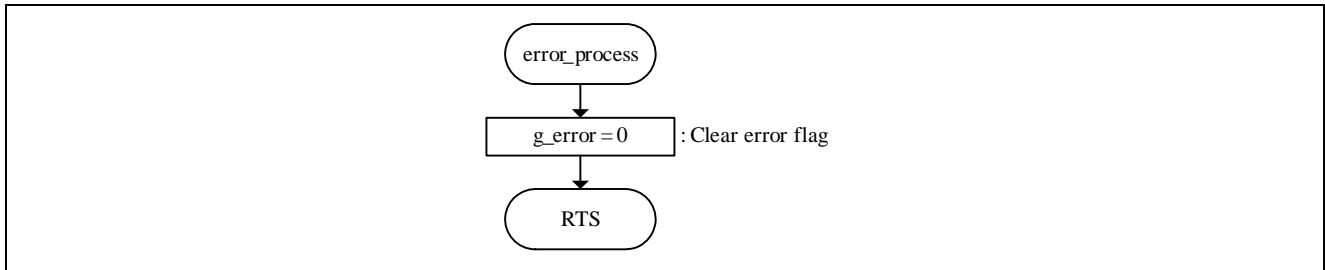
Figure 14   Error Processing Flow Chat

4. Sample Program

```c
/****************************************************************************
*                                                                          *
*    File Name: main.c                                                      *
*    Contents  : Main program of serial I/O sample program                 *
*    Copyright, 2005 RENESAS SYSTEM SOLUSTIONS (BEIJING) CO.,LTD.           *
*                                                                          *
*    Version:   1.00                                                        *
*                                                                          *
****************************************************************************/

#include "sfr_7542.h"
#include "sio.h"

void main(void)
{
    __asm("SEI");                  /* Interrupt Disable */

    /* Delay for external OSCI stabilization */
    PREX = 50 - 1;                 /* 2MHz(On-chip) * 1/16 * 1/50 * 1/25 = 10ms */
    TX = 25 - 1;
    IREQ2.3 = 0;                   /* Clear Timer X interrupt request bit */

    sfr_init();

    while (IREQ2.3 == 0){}         /* Wait for Timer X underflow */
    IREQ2.3 = 0;                   /* Clear Timer X interrupt request bit */

    CPUM = 0x00;                   /* Change to main clock */

    /* Setting main cycle timer */
    PREX = 250 - 1;                /* 8MHz * 1/16 * 1/250 * 1/40 = 20ms */
    TX = 40 - 1;

    while(1)
    {
        __asm("CLI");              /* Enable interrupt */

        while (IREQ2.3 == 0){} /* Waiting for Timer X underflow */
        IREQ2.3 = 0;               /* Clear Timer X interrupt request bit */

        if (g_error != 0)
        {
            error_process();       /* Error processing */
        }

        key_check();               /* Check keys */
        mode_func();               /* Mode processing */
    }
}

/****************************************************************************
; Name:        sfr_init
```
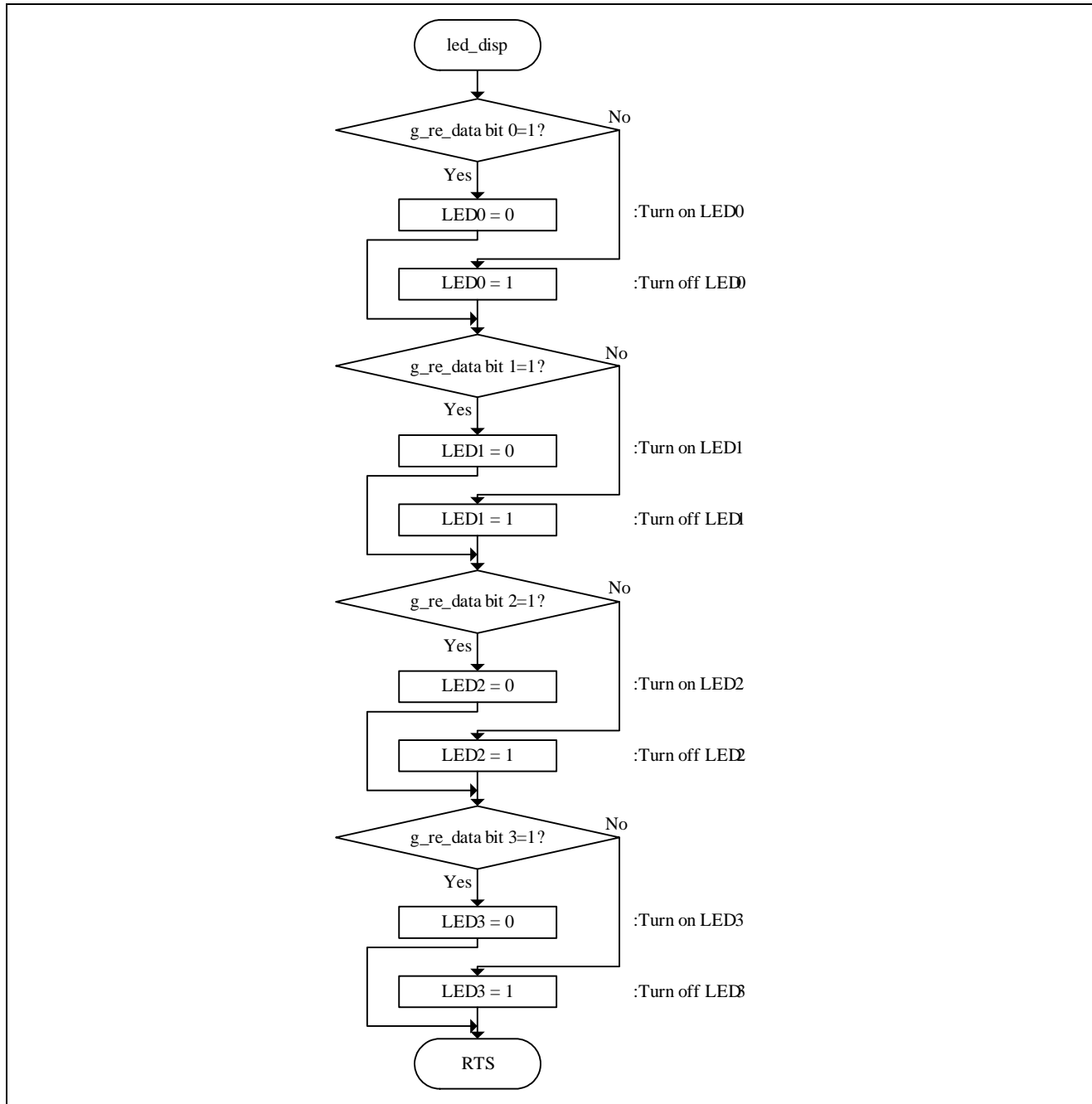
```
; Parameter:    None
; Return:       None
; Description: Initial setting of SFR registers
;****************************************************************************/


void sfr_init(void)
{
    DCCR = 0x1c;                /* P0 P3 drive capacity configuration for LED */
    P0 = 0x08;                  /* P03 = H(LED3) */
    P0D = 0xff;                 /* PD03 = Output(LED3), unused pins set to output */
    P1D = 0x1f;                 /* Unused pins output */
    P2D = 0xfe;                 /* PD20 = Input(AD0), unused pins set to output */
    P3 = 0x07;                  /* P30-2 = H(LED0,1,2) */
    P3D = 0x6f;                 /* P30-2 = Output(LED0,1,2), P37 = Input (SW1) */

    INTEDGE = 0xe0;             /* P04, P06 Key-on Wakeup Disable;
                                   INT0 Falling Edge Active */

    UART1CON = 0x02;            /* Even parity checking enable */
    BRG1 = 52 - 1;              /* 8MHz * 1/16 * 1/52 = 9615 bps */
    SIO1CON = 0xb0;             /* Serial I/O 1 enable;
                                   Transmit enable; Receive enable */

    TABM   = 0x0a;              /* Stop Timer A and Timer B */

    IREQ1  = 0;                 /* Clearing interrupt request bit */
    IREQ2  = 0;

    ICON1  = 0x11;              /* INT0 interrupt enable;
                                   Serial I/O1 receive interrupt enable */
}


/****************************************************************************
; Name:        mode_func
; Parameter:   None
; Return:      None
; Description: Mode processing
;****************************************************************************/
void mode_func(void)
{
    if (g_mode == TRANSMITTER)
    {
        sio1_transmit();          /* Call transmit function */
    }

    led_disp();                   /* Call display function */
}


/****************************************************************************
; Name:        key_check
; Parameter:   None
; Return:      None
; Description: Confirm pressed key twice and change g_mode
;****************************************************************************/
```

```
void key_check(void)
{
    switch (g_key_state)
    {
        case 0:
            break;

        case 1:                     /* 20ms delay */
            g_key_state++;
            break;

        case 2:
            if (SW1 == SW_ON)      /* Confirm key state */
            {
                g_key_state++;
            }
            else
            {
                g_key_state = 0;
            }
            break;

        case 3:
            if (SW1 == SW_ON)      /* Confirm key state again */
            {
                g_mode = TRANSMITER;

                g_key_state++;
            }
            else
            {
                g_key_state = 0;
            }
            break;

        case 4:
            if (SW1 == SW_OFF)     /* If key is released */
            {
                g_key_state = 0;   /* Set to no key pushed state */
            }
            break;

        default:
            break;
    }
}


/*******************************************************************************
; Name:        led_disp
; Parameter:   None
; Return:      None
; Description: Display receive data by LED
;*******************************************************************************/
void led_disp(void)
```

```
{
    if (g_re_data & 0x01)
    {
        LED0 = 0;                    /* Turn on LED0 */
    }
    else
    {
        LED0 = 1;                    /* Turn off LED0 */
    }

    if (g_re_data & 0x02)
    {
        LED1 = 0;                    /* Turn on LED1 */
    }
    else
    {
        LED1 = 1;                    /* Turn off LED0 */
    }

    if (g_re_data & 0x04)
    {
        LED2 = 0;                    /* Turn on LED2 */
    }
    else
    {
        LED2 = 1;                    /* Turn off LED2 */
    }

    if (g_re_data & 0x08)
    {
        LED3 = 0;                    /* Turn on LED3 */
    }
    else
    {
        LED3 = 1;                    /* Turn on LED3 */
    }

}

/******************************************************************************
; Name:       sio1_transmit
; Parameter:  None
; Return:     None
; Description: Transmit g_tr_data by Serial I/O 1
;******************************************************************************/
void sio1_transmit(void)
{
    if (SIO1STS.0 == 1)             /* Check transmit buffer full flag */
    {
        TB1RB1 = g_tr_data;

        g_tr_data--;                /* Reduce transmit data by 1 */
        if (g_tr_data == 0xff)
        {
```

```
            g_tr_data = 0x0f;
        }


        g_mode = RECEIVER;          /* When complete return to receive mode */
    }
}


/******************************************************************************
; Name:        error_process
; Parameter:   None
; Return:      None
; Description: Receive error processing
;******************************************************************************/
void error_process(void)
{
    /* User define */

    g_error = 0;                    /* Clear error flag */
}



/******************************************************************************
; Name:        INT0_ISR
; Parameter:   None
; Return:      None
; Description: INT0 interrupt handler
;******************************************************************************/
interrupt void INT0_ISR(void)
{
    if (g_key_state == 0)        /* If no key pressed */
    {
        g_key_state = 1;
    }
}

/******************************************************************************
; Name:        SIO1RE_ISR
; Parameter:   None
; Return:      None
; Description: Serial I/O 1 receive interrupt handler
;******************************************************************************/
interrupt void SIO1RE_ISR(void)
{
    if (SIO1STS.6 == 0)          /* Check summing error bit */
    {
        g_re_data = TB1RB1;
    }
    else
    {
        g_error = 1;                 /* Set error flag */
        IO1STS.6 = 0;                /* Clear summing error bit */
        __asm("LDA $18");            /* Read receive buffer for clearing RBF flag*/
    }
}
```

```
/*****************************************************************************
*                                                                           *
*   File Name  : sio.h                                                       *
*   Contents   : Definition of sample program                               *
*   Copyright, 2005 RENESAS SYSTEM SOLUSTIONS (BEIJING) CO.,LTD.             *
*                                                                           *
*   Version: 1.00                                                            *
*                                                                           *
*****************************************************************************/


#ifndef SIO_H
#define SIO_H


/* Definition of globle variable */

unsigned char g_mode;                   /* Mode flag */
unsigned char g_error;                  /* Error flag */
unsigned char g_tr_data = 0x0f;         /* Transmit data buffer (Initialized as 0x0f) */
unsigned char g_re_data;                /* Receive data buffer */
unsigned char g_key_state;         /* Key state */


/* Declaration of function prototype */

extern void sfr_init(void);             /* SFR initialize routine */
extern void mode_func(void);            /* Mode processing */
extern void key_check(void);            /* Check key input */
extern void led_disp(void);             /* LED control */
extern void sio1_transmit(void);        /* Transmit routine */
extern void error_process(void);        /* Error processing */


/* Definition of macros */
#define SW_ON          0
#define SW_OFF         1

#define RECEIVER       0
#define TRANSMITTER    1

#define LED0           P3_0
#define LED1           P3_1
#define LED2           P3_2
#define LED3           P0_3

#define SW1            P3_7

#endif                                  /* SIO_H */
```

5. Reference

Renesas web-site
http://www.renesas.com/

Contact for Renesas technical support
E-mail: csc@renesas.com

Data Sheet
7542 Group Rev.3.00
(Use the latest version on the home page:http://www.renesas.com)

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Dec.28.05	—	First edition issued

Keep safety first in your circuit designs!

1. 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

1. 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.
2. 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.
3. 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 rising 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).
4. 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.
5. 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.
6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
7. 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.
8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.