致尊敬的顾客

关于产品目录等资料中的旧公司名称

NEC电子公司与株式会社瑞萨科技于2010年4月1日进行业务整合（合并），整合后的新公司暨“瑞萨电子公司”继承两家公司的所有业务。因此，本资料中虽还保留有旧公司名称等标识，但是并不妨碍本资料的有效性，敬请谅解。

瑞萨电子公司网址：http://www.renesas.com

2010年4月1日
瑞萨电子公司

【发行】瑞萨电子公司（http://www.renesas.com）

【业务咨询】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.

H8/300L Super Low Power（SLP）系列

对于 LCD 和串行端口的 printf 函数

内容

本应用说明叙述 printf 函数的写法，说明从库函数调用的方法和用户建立的方法。另外，也说明将信息的输出目标设定为 SIM IO 窗口、LCD 和串行端口的方法。

要点

和具有标准输入（键盘）和标准输出（显示器）设备的 PC 不同，嵌入式系统需要开发者定义输入源和输出目标。

以 printf 函数为例，嵌入式系统开发者必须定义此函数的输出目标。作为被广泛使用的输出，有 LCD 显示屏和 PC 的超级终端（经由设备的串行端口）。

本应用说明使用 5 个 HEW2.1 工程文件，说明面向 SLP H8/38024F 的以下 5 类例子。操作例是使用 SLP CPU 电路板（ALE300L）和应用电路板的例子。

1. Simulated I/O
2. LCD 显示屏的 printf（使用标准库函数）
3. 串行端口的 printf（使用标准库函数）
4. LCD 显示屏的基本的 printf（使用用户建立的程序）
5. 串行端口的基本的 printf（使用用户建立的程序）

动作确认器件

H8/38024

目录

1. printf 函数的使用方法

printf 是嵌入式系统开发者广泛使用的函数，具有将信息输出到外部的功能，主要的使用方法有以下两种：

(1) 提供用户界面（例：通过血压仪在 LCD 显示屏显示血压值）

(2) 提供调试手段（例：将实际的 ADC 值从串行端口发送到 PC 的超级终端）

【注】 以下的例子是使用 HEW2.1（H8 TINY/Super Low Power 工具链）创建的。不能用以前版本的 HEW 存取在此使用的工程文件。如果所使用的版本不同，最简单的方法就是通过所使用的 HEW 建立新的工程，并在该新工程目录中替换需要的 C 程序和头文件。

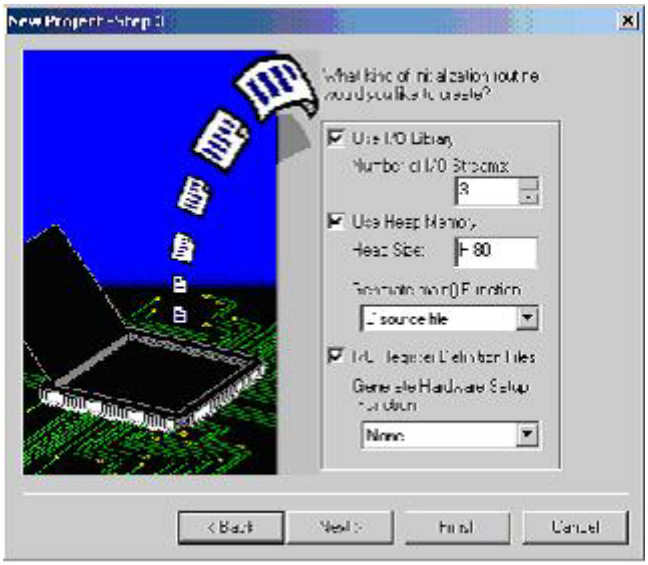
2. Simulated I/O

在硬件准备完毕之前，能通过 HEW 仿真器开始工作。为了进一步提高调试环境的效率，HEW 仿真器导入了 Simulated I/O。Simulated I/O 能使开发者将结果（调试信息）输出到 HEW 的调试（SIM IO）窗口。即 printf 函数能将信息显示在此窗口。

HEW 工程生成程序自动进行 SIM I/O 功能的设定（在此例中，使用 H8S、 H8/300 标准工具链创建工程）。免费的 H8 TINY/Super Low Power 工具链没有仿真器功能。

确认 Simulated I/O 结果的方法如下所示：

(1) 在工程生成的步骤 3，选择 **Use I/O Library** 选项和 **Use heap memory** 选项，I/O 流数为 3 不变。



(2) 由此，生成和更改了以下的文件：

a. Lowsrc.c（低级标准输入/输出功能）

b. Lowlev.src（包含 charget 和 charput 的输出目标– PC SIM IO 窗口）

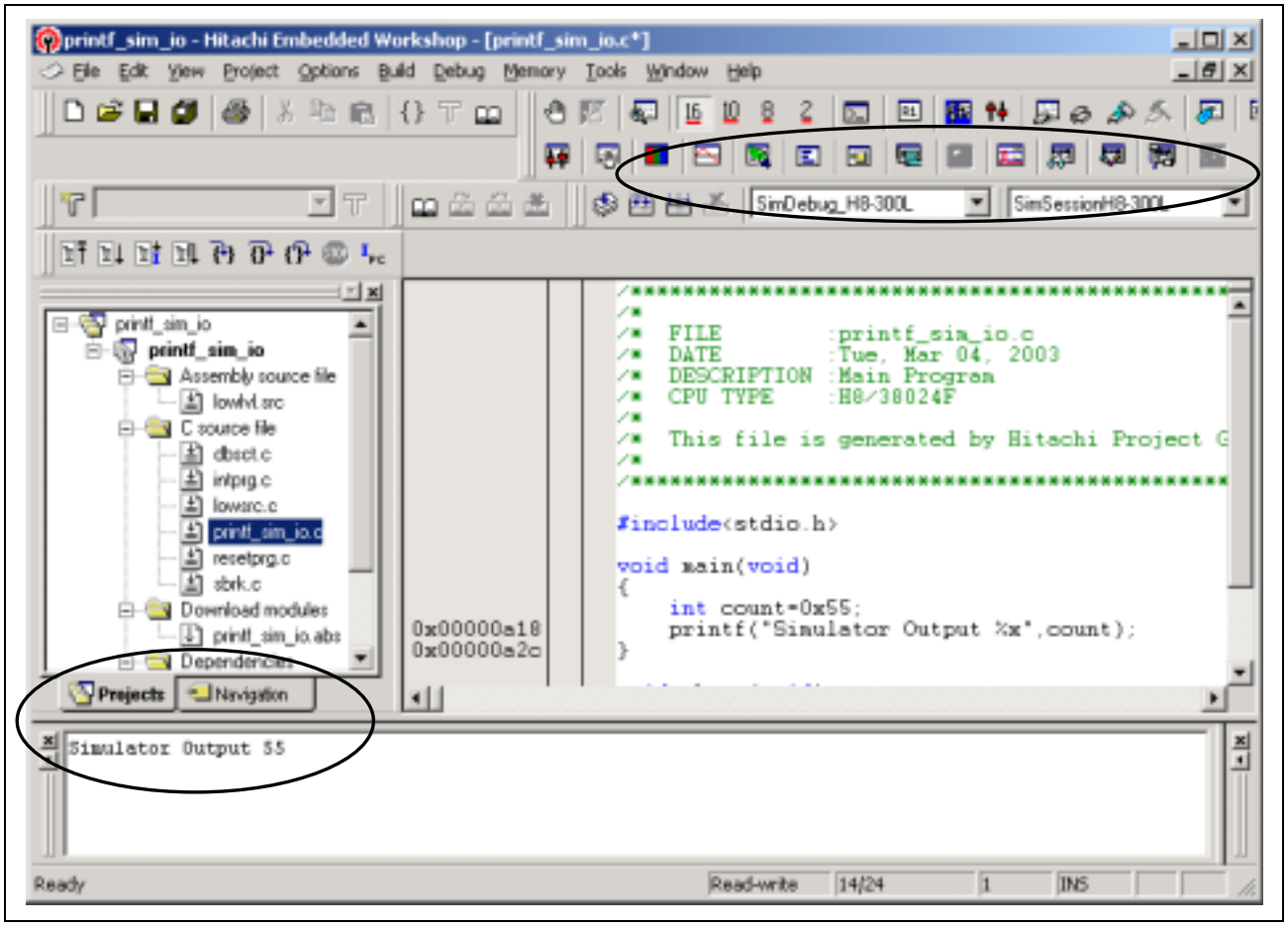
c. Resetprg.c（追加函数_INIT_IOLIB()和_CLOSEALL()的调用，初始化标准 I/O）

d. Sbrk.c（分配堆存储器）

(3) 将 printf()追加到主程序（必须追加#include <stdio.h>）。
(4) 将调试会话的默认值从“Debug”更改为“SimDebug_H8-300L”。
(5) **创建**工程（F7）。
(6) 打开 Option/**Debug Setting** 窗口，如果将会话设定为“SimSessionH8-300L”，就自动设定以下内容：
   a. 将目标设定为“H8/300L Simulator”
   b. 将调试格式的默认值设定为“Elf/Dwarf2”
   c. 作为在步骤 v 的编译文件，追加下载模块

(7) 将会话更改为“SimSession H8-300L”。
(8) 打开 Option/ simulator/ **Simulator Memory Resources**。
   a. 通过存储器映像，指定可读/可写，追加能使用的存储器。作为系统内存资源，显示所分配的资源（HEW 自动设定）。

(9) 打开 Option/ simulator/ **Simulator System**，将系统调用地址设定为允许（HEW 自动设定）。
(10) 打开 Debug/**Download Modules**，装入文件（Debug setting 预先设定的文件）。
(11) 打开 View/ **Simulated IO**。
(12) 点击 Debug/**Reset Go**。
(13) 确认 Simulated I/O 窗口的信息。



详细步骤和说明请参照 HEW 的在线用户手册。

HEW 生成的复位例程如下所示：

```
__entry(vect=0) void PowerON_Reset(void)
{
     set_imask_ccr(1);
    _INITSCT();
//  _CALL_INIT();    // Remove the comment when you use global class object
    _INIT_IOLIB();   // Use SIM I/O
//  errno=0;      // Remove the comment when you use errno
//  srand(1);     // Remove the comment when you use rand()
//  _s1ptr=NULL;      // Remove the comment when you use strtok()
//  HardwareSetup(); // Remove the comment when you use Hardware Setup
    set_imask_ccr(0);

    main();

    _CLOSEALL();     // Use SIM I/O
//  _CALL_END();     // Remove the comment when you use global class object
    sleep();
}
```

HEW 生成的 Lowlev.src 如下所示，将字符发送到 SIM IO。

```
        .EXPORT    _charput
        .EXPORT    _charget

SIM_IO:  .EQU      H'0000

        .SECTION   P,CODE,ALIGN=2
;---------------------------------------------------------------------
;  _charput:
;---------------------------------------------------------------------
_charput:
        MOV.B      R0L,@IO_BUF
        MOV.W      #H'0102,R0
        MOV.W      #IO_BUF,R1
        MOV.W      R1,@PARM
        MOV.W      #PARM,R1
        JSR        @SIM_IO
        RTS
;---------------------------------------------------------------------
;  _charget:
;---------------------------------------------------------------------
_charget:
        MOV.W      #H'0101,R0
        MOV.W      #IO_BUF,R1
        MOV.W      R1,@PARM
        MOV.W      #PARM,R1
        JSR        @SIM_IO
        MOV.B      @IO_BUF,R0L
        RTS
;---------------------------------------------------------------------
;  I/O Buffer
;---------------------------------------------------------------------
        .SECTION   B,DATA,ALIGN=2
PARM:    .RES.W     1
IO_BUF:  .RES.B     1
        .END
```

3. charput 函数和 charget 函数的变更

由上例可知， printf()将信息输出到 SIM IO 窗口时所需的函数全部由 HEW 函数生成程序生成。要将信息输出到串行端口和 LCD 等其他设备时，更改 lowsrc.src 文件的 charput 函数。能将这 2 个函数用 C 语言记述并且包含于其他的.c 文件中。

举例说明以下 2 种输出手段：

(1) 串行端口

(2) LCD

3.1 输出到串行端口的 printf

3.1.1 程序的说明

说明使用 SLP 应用电路板的串行端口 3（SCI-3）的例子。

main 例程初始化通用 I/O 和 SCI-3，其次 printf 函数使用 charput 函数输出一连串的字符，最后 charput 函数经由串行端口输出数据。

“no_float.h”必须声明在“stdio.h”前，由此缩小 printf 函数的程序大小（不使用浮点格式的情况）。

```
#include <no_float.h>
#include <stdio.h>
#include "iodefine.h"
#include <machine.h>

static const char string[] = {"¥n¥n¥rCan putstr too!"};

void main(void)
{
    int count=0;

    init_io();
    init_sci();

    printf("¥n¥n¥n¥rDemonstration of printf function");
    printf("¥n¥rCounting = ");

    for (count=0;count<10;count++)
        printf(" %d",count);

    PutStr((char *)string);
}
```

以下详细说明 main 函数调用的 3 个函数：

1. void init_io(void); //通用初始化程序

```
void init_io(void)
{
    P_IO.PCR3.BYTE = 0x00;     //P37..P31 : inputs
    P_IO.PUCR3.BYTE = 0x00;    //Turn off the MOS pull-up

    //PMR3 : |AEVL|AEVH|---|---|---|TMOFH|TMOFL|---|
    //AEVL = 0 : P37 as I/O
    //AEVH = 0 : P36 as I/O
    //TMOFH = 0: P32 as I/O
    //TMOFL = 0: P31 as I/O
    P_IO.PMR3.BYTE = 0x00;

    P_IO.PCR4.BYTE = 0xF8;     //P40 is connected to keypad 0

    //PMR2 : |---|---|POF1|---|---|---|---|IRQ0| : |1|1|0|1|1|0|0|1|
    //Bits 7, 6, 4 and 3 are reserved and always read as 1 and cannot be
    //modified
    //POF1 = 0 (initial value)
    //Only 0 can be written to reserved bits 2 & 1.
    //IRQ0 = 1 : functions as IRQ0_N input pin
    P_IO.PMR2.BYTE = 0xD9;

    //PMR9 : |---|---|---|---|PIOFF|---|PWM2|PWM1|
    //PIOFF = 0 : large-current port step-up circuit is turned on
    //PWM1 = PWM2 = 0 : P90 and P91 functions as P_IO output pin
    P_IO.PMR9.BYTE = 0xF0;

    //PMRB : |---|---|---|---|IRQ1|---|---|---|
    //IRQ1 = 0 : PB3 functions as I/O or AN3
    //Bits 7 to 4 and 2 to 0 are reserved and always read as 1.
    P_IO.PMRB.BYTE = 0xF7;

#if ALE300L_38024||ALE300L_3802
    init_sci();
#endif
```

```
    //IEGR : |---|---|---|---|---|---|IEG1|IEG0| : |1|1|1|0|0|0|0|0|
    //Bits 7 to 5 are reserved; they are always read as 1 and cannot be
   //modified
    //Bits 4 to 2 are reserved; only 0 can be written to these bits
    //IEG0 = 0 : Falling edge of IRQ0_N is detected
    P_SYSCR.IEGR.BYTE = 0xE0;

    //IENR1 : |IENTA|---|IENWP|---|---|IENEC2|IEN1|IEN0| : |0|0|0|0|0|
    //IENTA = 0 : Disable Timer A interrupt request
    //Bit 6 is reserved; only 0 can be written to it
    //IENWP = 0 : Disable WKP7_N TO WKP0_N interrupt requests
    //Bits 4 and 3 are reserved; only 0 can be written to these bits
    //IENEC2 = 0 : Disable IRQAEC interrupt request
    //IEN1 = 0 : Disable interrupt request from IRQ1_N
    //IEN0 = 1 : Enable interrupt request from IRQ0_N
    P_SYSCR.IENR1.BYTE = 0x01;
    P_SYSCR.IENR2.BYTE = 0x10;
    set_imask_ccr(0);
}
```

2. void init_sci(void); // 将 SCI-3 初始化为 2400bps、8 位、1 个停止位、无奇偶校验

```
void init_sci(void)
{   unsigned char temp = 0;

    //SCR3 : |TIE|RIE|TE|RE|MPIE|TEIE|CKE1|CKE0|
    //TIE : Transmit interrupt enable
    //RIE : Receive interrupt enable
    //TE  : Transmit enable
    //RE  : Receive enable
    //MPIE : Multiprocessor interrupt enable
    //TEIE : Transmit end interrupt enable
    //CKE1 : Clock enable 1
    //CKE0 : Clock enable 0
    //CKE1 = CKE0 = 0
    //asynchronous mode, internal clock source, SCK32 functions as I/O port
    P_SCI3.SCR3.BYTE = 0x30;

    //SMR : |COM|CHR|PE|PM|STOP|MP|CKS1|CKS0| : |0|0|0|0|0|0|0|0|
    //COM : Communication Mode  : 0 : asynchronous mode
    //CHR : Character Length    : 0 : character length = 8 bits
    //PE : Parity Enable     : 0 : parity bit addition and checking disabled
    //PM  : Parity Mode  : 0 : even parity (no effect since parity is already
   //disabled)
    //STOP: Stop Bit Length     : 0 : 1 stop bit
    //MP  : Multiprocessor Mode : 0 : multiprocessor communication function
   //disabled
    //|CKS1|CKS0| : Clock Select: |0|0| : clock source for baud rate generator
   //= clk
    P_SCI3.SMR.BYTE = 0x00;

    //Bit rate = 19200 bps, n = 0, N = 64 // MODIFY TO 2400bps
    P_SCI3.BRR = 64;

    //SPCR : |---|---|SPC32|---|SCINV3|SCINV2|---|---| : |1|1|1|0|0|0|0|0|
    //SPC32 = 1 : P42 functions as TXD32 output pin
    //need to set TE bit in SCR3 after setting this bit to 1
    //SCINV3 = 0 : TXD32 output data is not inverted
    //SCINV2= 0 : RXD32 input data is not inverted
    //Bits 7 and 6 are reserved and always read as 1
    //Bits 4, 1 and 0 are reserved and only 0 can be written to these bits
    P_SCI3.SPCR.BYTE = 0xE0;

   //SSR  : |TDRE|RDRF|OER|FER|PER|TEND|MPBR|MPBT|
    //TDRE : transmit data register empty
    //RDRF : receive data register full
    //OER  : overrun error
    //FER  : framing error
    //PER  : parity error
    //TEND : transmit end
    //MPBR : Multiprocessor bit receive
    //MPBT : Multiprocessor bit transfer
    P_SCI3.SSR.BYTE = 0x84;             //Initialise upon reset to 0x84
}
```

3. void charput(char outputchar) //将数据输出到串行端口 3

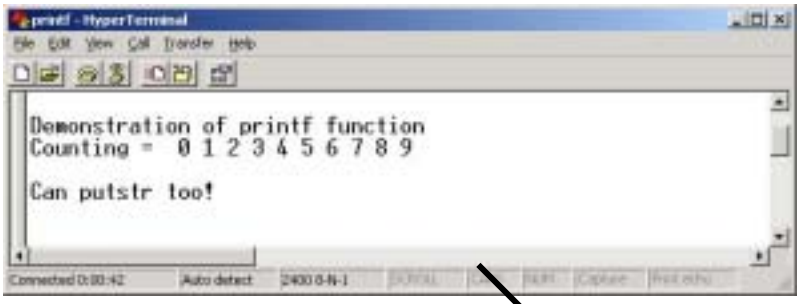
```
void charput(char OutputChar)            //Serial Port
{
  while ((P_SCI3.SSR.BIT.TDRE) == 0);

  P_SCI3.TDR = OutputChar;
  P_SCI3.SSR.BIT.TDRE = 0;
}
```

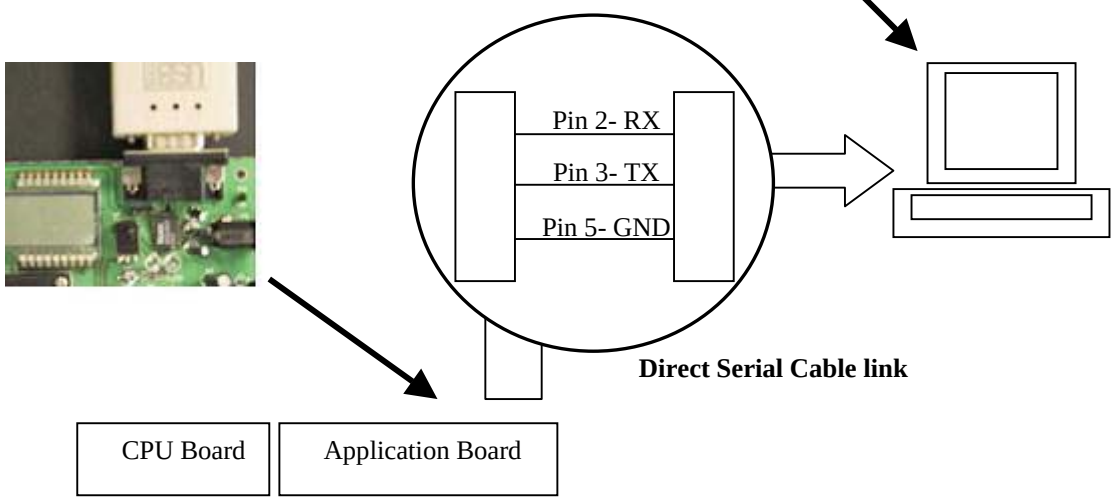
3.1.2 硬件的设定

能设定 PC 窗口超级终端，显示此信息。使用串行电缆（直接连接型电缆）连接应用电路板和 PC 串行端口。超级终端的设定如下图所示：

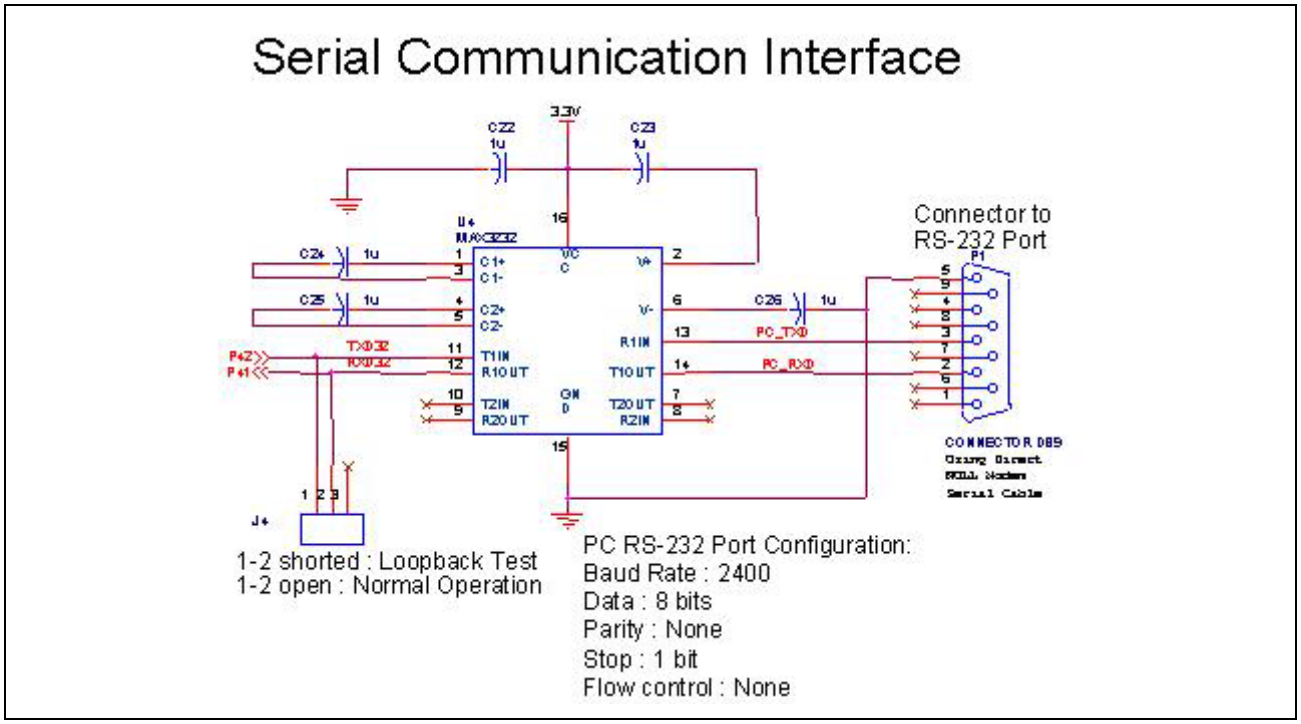
信息的输出如下图所示：



硬件设定如下：



为了取得 MCU 和 PC 的接口，应用电路板需要 RS232 驱动器：



3.2 输出到 LCD 的 printf

3.2.1 程序的说明

说明使用 SLP 应用电路板的 LCD 的例子。

main 例程能在初始化通用 I/O 和 LCD 后调用标准库的 printf 函数。此时，charput 函数将字符输出到 LCD（1 行×7 个字符）到。全局变量 *position* 决定显示在 LCD 的字符位置。

```
#include <no_float.h>
#include <stdio.h>

#include "iodefine.h"
#include "printf_lcd.h"
#include <machine.h>

unsigned int position=7;

void main(void)
{
  char temp1 ='e';
  int  temp2 =15; //0xF

  init_io();
  init_lcd();

  printf("HI %c %x", (BYTE)temp1, (DWORD)temp2);
 }
```

程序例使用 4 个主要函数：

1. void init_io(void); //通用初始化例程（同 3.1 章的函数）

2. void init_lcd(void) //LCD 的初始化

```
void init_lcd(void)
{
    unsigned char temp_a;
    unsigned char *dest;

    //clear LCD RAM
    dest = (unsigned char *)0xF740;
    for (temp_a = 0 ; temp_a < 16 ; temp_a++)
    {
        *dest++ = 0;
    }

    //LPCR : |DTS1|DTS0|CMX|---|SGS3|SGS2|SGS1|SGS0| : |1|1|0|0|0|1|1|0|
    //|DTS1|DTS0| = |1|1| : 1/4 duty
    //|CMX | = 0
    //Bit 4 is reserved; only 0 can be written to this bit
    //|SGS3|SGS2|SGS1|SGS0| = |1|0|0|0| : Use SEG1 to SEG32
    P_LCD.LPCR.BYTE = 0xC8;    //1/4 duty cycle

    //LCR : |---|PSW|ACT|DISP|CKS3|CKS2|CKS1|CKS0| : |1|1|1|1|1|1|1|1|
    //Bit 7 is reserved; always read as 1 and cannot be modified
    //PSW = 1 : LCD drive power supply on
    //ACT = 1 : LCD controller/driver operates
    //DISP = 1 : LCD RAM data is displayed
    P_LCD.LCR.BYTE = 0xFF;     //display is faint

    //LCR2 : |LCDAB|---|---|---|---|---|---|---|
    //LCDAB : 0 : drive using A waveform
    //Bits 6 and 5 are reserved; always read as 1 and cannot be modified
    //Bits 4 to 0 are reserved; only 0 can be written to these bits
    P_LCD.LCR2.BYTE = 0x60;
}
```

3. void Display_number(…) // 在 LCD 显示数值，LCD 只能显示 7 个字符。

```
void display_number(unsigned char digit, unsigned char number, unsigned char decimal_point)
{
    unsigned short   *dest;

    switch(digit)
    {
        case 0:       dest = (unsigned short *)0xF740;
                  break;
        case 1:       dest = (unsigned short *)0xF742;
                  break;
        case 2:       dest = (unsigned short *)0xF744;
                  break;
        case 3:       dest = (unsigned short *)0xF746;
                  break;
        case 4:       dest = (unsigned short *)0xF748;
                  break;
        case 5:       dest = (unsigned short *)0xF74A;
                  break;
        case 6:       dest = (unsigned short *)0xF74C;
                  break;
        case 7:       dest = (unsigned short *)0xF74E;
                  break;
        default:      break;
    }

    if (decimal_point)
        *dest = (unsigned short)(lcd_number_data[number] | 0x0800);
    else
        *dest = lcd_number_data[number];
}
```

4. void charput(char outputchar)　　//调用 display_number()，显示字符。

```
void charput(char OutputChar)
{
    display_number(position, OutputChar, 0);
    position--;
}
```

5. printf_lcd.h // 搜索转换显示在 LCD 的字符表。

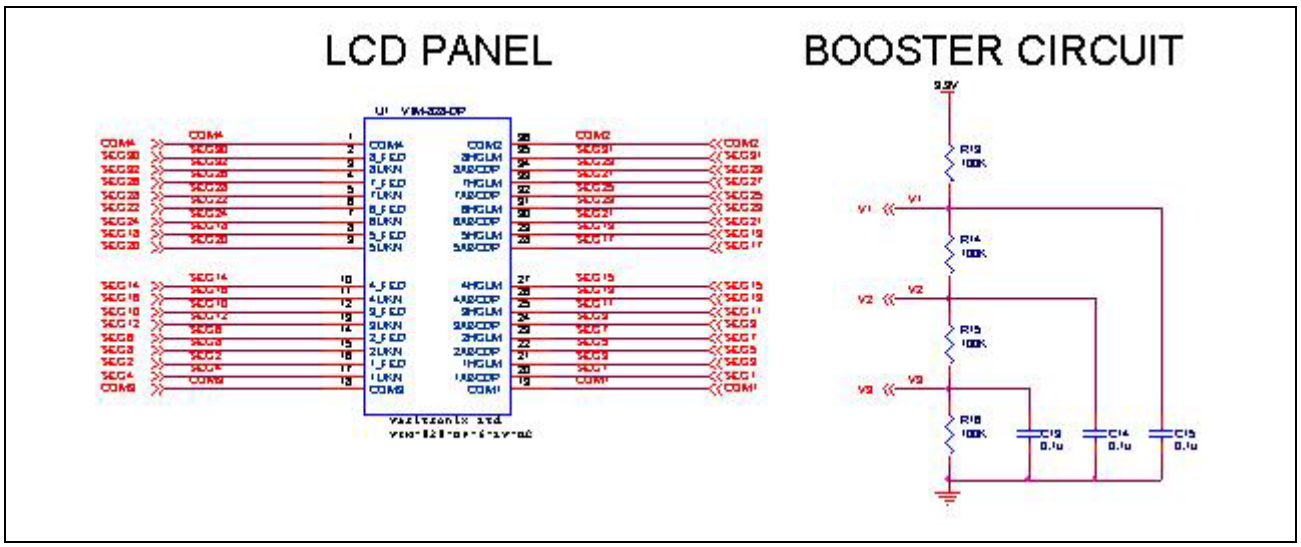
```
//for 1/4 duty cycle
const unsigned shortlcd_number_data[128]
= {               //ASCII
                  0x0039,     //  00.   'NUL'    :Display
                  0x0039,     //  01.   'SOH'    :Display
                  0x0039,     //  02.   'STX'    :Display
                  0x0039,     //  03.   'ETX'    :Display
                  0x0039,     //  04.   'EOT'    :Display
                  0x0039,     //  05.   'ENQ'    :Display
                  0x0039,     //  06.   'ACK'    :Display
                  0x0039,     //  07.   'BEL'    :Display
                  0x0039,     //  08.   'BS'     :Display
                  0x0039,     //  09.   'HT'     :Display
                  0x0039,     //  0A.   'LF'     :Display
                  0x0039,     //  0B    'VT'     :Display
                  0x0039,     //  0C.   'FF'     :Display
                  0x0039,     //  0D.   'CR'     :Display
                  0x0039,     //  0E.   'SO'     :Display
                  0x0039,     //  0F.   'SI'     :Display
                  0x0039,     //  10.   'DLE'    :Display
                  0x0039,     //  11.   'DC1'    :Display
                  0x0039,     //  12.   'DC2'    :Display
                  0x0039,     //  13.   'DC3'    :Display
                  0x0039,     //  14.   'DC4'    :Display
                  0x0039,     //  15.   'NAK'    :Display
                  0x0039,     //  16.   'SYN'    :Display
                  0x0039,     //  17.   'ETB'    :Display
                  0x0039,     //  18.   'CAN'    :Display
                  0x0039,     //  19.   'EM'     :Display
                  0x0039,     //  1A.   'SUB'    :Display
                  0x0039,     //  1B.   'ESC'    :Display
                  0x0039,     //  1C.   'FS'     :Display
                  0x0039,     //  1D.   'GS'     :Display
                  0x0039,     //  1E.   'RS'     :Display
                  0x0039,     //  1F.   'US'     :Display
                  0x0000,     //  20.   'SP'     :Space, all segment off
                  0x0039,     //  21.   '!'      :Display
                  0x2200,     //  22.   '"'      :"
                  0x0039,     //  23.   '#'      :Display
                  0xA55A,     //  24.   '$'      :$
                  0x0039,     //  25.   '%'      :Display
                  0x0039,     //  26.   '&'      :Display
                  0x0010,     //  27.   '''      :'
                  0x0039,     //  28.   '('      :Display
                  0x0039,     //  29.   ')'      :Display
                  0x00E7,     //  2A.   '*'      :*
                  0x005A,     //  2B.   '+'      :+
                  0x0039,     //  2C.   ','      :Display
                  0x0042,     //  2D.   '-'      :-
                  0x0800,     //  2E.   '.'      :.
                  0x0024,     //  2F.   '/'      :/
```

```
 0xE724,     //  30.   '0'      :0
0x0600,      //  31.   '1'      :1
0xC342,      //  32.   '2'      :2
0x8742,      //  33.   '3'      :3
0x2642,      //  34.   '4'      :4
0xA542,      //  35.   '5'      :5
0xE542,      //  36.   '6'      :6
0x0700,      //  37.   '7'      :7
0xE742,      //  38.   '8'      :8
0x2742,      //  39.   '9'      :9
0x0039,      //  3A.   ':'      :Display
0x0039,      //  3B.   ';'      :Display
0x00A0,      //  3C.   '<'      :<
0x8100,      //  3D.   '='      :=
0x0005,      //  3E.   '>'      :>
0x0039,      //  3F.   '?'      :Display
0x0039,      //  40.   '@'      :Display
0x6742,      //  41.   'A'      :A
0xE442,      //  42    'B'      :b
0xE100,      //  43.   'C'      :C
0xC642,      //  44.   'D'      :D
0xE142,      //  45.   'E'      :E
0x6142,      //  46.   'F'      :F
0xE540,      //  47.   'G'      :G
0x6642,      //  48.   'H'      :H
0x8118,      //  49.   'I'      :I
0xC600,      //  4A.   'J'      :J
0x60A2,      //  4B.   'K'      :K
0xE000,      //  4C.   'L'      :L
0x6621,      //  4D.   'M'      :M
0x6681,      //  4E.   'N'      :N
0xE700,      //  4F.   'O'      :O
0x6342,      //  50.   'P'      :P
0xE780,      //  51.   'Q'      :Q
0x63C2,      //  52.   'R'      :R
0xA542,      //  53.   'S'      :S
0x0118,      //  54.   'T'      :T
0xE600,      //  55.   'U'      :U
0x0681,      //  56.   'V'      :V
0x6684,      //  57.   'W'      :W
0x00A5,      //  58.   'X'      :x
0x0029,      //  59.   'Y'      :Y
0x8124,      //  5A.   'Z'      :Z
0xE100,      //  5B.   '['      :[
0x0081,      /*  5C.   '¥'      :¥     */
0x8700,      //  5D    ']'      :]
0x0084,      //  5E.   '^'      :^
0x0000,      //  5F.   ' '      :
0x0001,      //  60.   '`'      :`
0xC742,      //  61.   'a'      :a
0xE442,      //  62    'b'      :b
0xC042,      //  63.   'c'      :c
0xC642,      //  64.   'd'      :d
```

```
                0xE142,     //  65.   'e'      :E
                0x6142,     //  66.   'f'      :F
                0xE540,     //  67.   'g'      :G
                0x6442,     //  68.   'h'      :h
                0x8118,     //  69.   'i'      :I
                0xC600,     //  6A.   'j'      :J
                0x60A2,     //  6B.   'k'      :K
                0xE000,     //  6C.   'l'      :L
                0x444A,     //  6D.   'm'      :m
                0x4442,     //  6E.   'n'      :n
                0xC442,     //  6F.   'o'      :o
                0x6342,     //  70.   'p'      :P
                0xE780,     //  71.   'q'      :Q
                0x63C2,     //  72.   'r'      :R
                0xA542,     //  73.   's'      :S
                0x0118,     //  74.   't'      :T
                0xE600,     //  75.   'u'      :U
                0x0681,     //  76.   'v'      :V
                0x6684,     //  77.   'w'      :W
                0x00A5,     //  78.   'x'      :x
                0x0025,     //  79.   'y'      :Y
                0x8124,     //  7A.   'z'      :Z
                0x0039,     //  7B.   '{'      :Display
                0x0018,     //  7C.   '|'      :|
                0x0039,     //  7D.   '}'      :Display
                0x0039,     //  7E.   '~'      :Display
                0x0039,     //  7F.   'DEL'    :Display
};
```

3.2.2 硬件的设定

将 LCD 显示屏直接连接到 SLP MCU：



应用电路板的 LCD 显示如下所示：

```
char k='e';
int j =15;

printf("Hi %c%x",(BYTE)k,(DWORD)j);
```

4. printf()库函数的缺点

printf()虽然好用，却是很复杂的库函数（关于函数的详细内容，请参照 H8 编译程序用户手册）。在 SLP MCU 使用时有以下缺点：

4.1 ROM 容量

因为 printf()函数必须处理 I/O 流、堆和许多参数，所以需要大量的 ROM 区（详细的比较请参照第 6 章）。

【注】 SLP H8/38024 的 ROM 容量最大为 32K 字节。

4.2 RAM 容量

因为 printf()需要使用堆存储器，所以只 printf 就被分配了 H’80 字节（不使用其他目的的堆的情况）。

【注】 SLP H8/38024 的 RAM 容量最大为 1K 字节。

5. 用户建立的 printf 函数 – Bprintf()

为解决上述问题，需要用户建立函数。本应用说明建立基本的 printf (Bprintf)函数，能根据应用的需要自定义此例。

和 2～3 章中说明的例子相比，Bprintf 函数不需要以下内容：

(1) Lowsrc.c
(2) Lowlev.src
(3) Resetprg.c 中的_INIT_IOLIB()和_CLOSEALL()函数调用
(4) sbrk.c

换言之，就是不需要 I/O 流和堆存储器，这正是用户建立函数的优点。

5.1 函数的使用方法

函数的原型如下：

```
BYTE Bprintf(const char *fmt, BYTE arg1, DWORD arg2);
```

此函数一般能和通常的 printf 函数一样调用，但是有如下限制：

(1) 显示字符数（20）
(2) 参数只有 2 个
(3) 第 1 参数是 BYTE 型，第 2 参数是 DWORD 型
(4) 支持的参数格式是%x、%c、%u

使用例：

(1) Bprintf(“Hello world”,0,0);
(2) Bprntf(“¥n¥rGet data = %x from address %x”, (BYTE)data, (DWORD)address);

5.2 函数的说明

和库的 printf()不同，Bprintf()是 printf 函数的简略版本，有以下限制：

(1) MAXCHARS ：指定字符串的长度，也是决定使用的堆栈深度的主要因素。
(2) 有 3 个 case 条件语句：参数的格式限制为%x、%c、%u 3 个。
(3) 变量 num ：参数的数量限制在 2 个。

Bprintf()函数调用 charput 函数，将信息发送到输出目标。

```
#define MAXCHARS      20
#define LEN   9

BYTE Bprintf(const char *fmt, BYTE arg1, DWORD arg2);
void itoab(char **buf, DWORD i, unsigned int base);


BYTE Bprintf(const char *fmt, BYTE arg1, DWORD arg2)
{
   DWORD u;
   BYTE num=0;                          // argument index
   BYTE index=0;                        // string index
   char buf[MAXCHARS];
   char *buf_ptr;

    buf_ptr = buf;

    // Rearranging output strings
   while (*fmt && index<(MAXCHARS-1))
   {    if (*fmt != '%')
           *buf_ptr++ = *fmt++;         // store string into buf
      else
      {   switch (*++fmt)               // if %, check what type
          {
              case 'x':                 // %x, hexadecimal unsigned number
                  if (num == 0)
                     u = (DWORD)arg1;
                  else if (num == 1)
                         u = (DWORD)arg2;
                     else
                         break;         //ignore > 2 arg
                     num++;
                  *buf_ptr++ = '0';
                  *buf_ptr++ = 'x';
                  b_itoab((char **)&buf_ptr, u, (unsigned int)16);
                  break;



               case 'u':                // %u, decimal unsigned number
                  if (num == 0)
                     u = (DWORD)arg1;
                  else if (num == 1)
                         u = (DWORD)arg2;
                     else
                         break;         //ignore > 2 arg
                     num++;
                  b_itoab((char **)&buf_ptr, u, (unsigned int)10);
                  break;
```

```
                case 'c':               // %c, a single character
                    if (num==0)
                        *buf_ptr++ = (char)arg1;
                    else if (num == 1)
                        *buf_ptr++ = (char)arg2;
                        else
                            break;      //ignore > 2 arg
                        num++;
                        break;

                default:
                        break;
                }// end switch
                fmt++;
            }// end else
        } //end while

        *buf_ptr = 0;                   // end of string indicator

        // Output rearranged string
        buf_ptr = buf;
        for (index = 0 ; *buf_ptr != (char)0 & index < MAXCHARS ; index++)
            charput(*buf_ptr++);        // output

        return(index);
    }
```

b_itoab()函数将整数转换为 ASCII 码，整数限于 16 进制或者 10 进制。

```
void b_itoab(char **buf, DWORD i, unsigned int base)
{
    BYTE index=0;
    DWORD rem;
    char conv[LEN];

    if (i == 0)
    {
        (*buf)[0] = '0';
        ++(*buf);
        return;
    }
    conv[index++] = 0;
    while (i)
    {
      rem = i % base;
      if (base == 10)
           conv[index++] = rem + '0';
      else if (base == 16)
      {
           if (rem < 10)
               conv[index++] = rem + '0';
           else
               conv[index++] = rem + 'A'- 0xA;
       }
       i /= base;
    }
    while (conv[--index])
    {
        (*buf)[0] = conv[index];
        ++(*buf);
    }
}
```

还能自定义 Bprintf()函数：

(1) 增加字符串的长度
(2) 追加参数的其他格式：%f、%3d、%o 等
(3) 增加参数的数量
(4) 增加或减小参数的长度（BYTE, WORD, DWORD）

6. 比较

为了比较程序大小，生成以下工程的映像文件（使用 Option/ Link Library/List）。为了简化比较，使用段 P 的大小评价函数的大小。在优化时，使用 debug 和 release 两者的设定。

	Debug	Release
printf LCD（不包含<no_float.h>）	20.4K 字节	14.9K 字节
printf LCD（包含<no_float.h>）	7.4K 字节	5.6K 字节
Bprintf LCD.	1.1K 字节	0.8K 字节

printf()的程序大小在使用浮点格式时约为 20.4K 字节；包含<no_float.h>时，被缩小到约 7.4K 字节。而用户建立函数时，程序大小进一步被缩小到 1.1K 字节，为 7 分之 1。必须再次注意 SLP 的最大 ROM 容量只有 32K 字节。

对于 RAM 容量，用户建立的 Bprintf()函数使用约 30 字节的堆栈（当 MAXCHAR=20 时），而库的 printf()函数使用 128（H’80）字节的堆存储器。

7. 总结

因为 SLP 的 ROM 容量小，并且在大部分目标应用程序中不需要复杂的 I/O 流，所以库的 printf()函数效率不高。因此推荐用户自定义 printf()函数。

公司主页和咨询窗口

有关本应用说明的技术方面的咨询请发邮件到下面的邮箱。

瑞萨科技公司主页　　http://www.cn.renesas.com
亚洲地区技术支持中心　　E-Mail：support.asia@renesas.com

修订记录

Rev.	发行日	修订内容	
		页	修订要点
1.00	2006.03.28	—	初版发行

Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corp. 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 Corp. 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 Corp. or a third party.
2. Renesas Technology Corp. 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 Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. 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 Corp. 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 Corp. by various means, including the Renesas Technology Corp. 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 Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. 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 Corp. or an authorized Renesas Technology Corp. 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 Corp. 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 Corp. for further details on these materials or the products contained therein.

注意

本文只是参考译文，前页所载英文版“Cautions”具有正式效力。

请遵循安全第一进行电路设计

1. 虽然瑞萨科技尽力提高半导体产品的质量和可靠性，但是半导体产品也可能发生故障。半导体的故障可能导致人身伤害、火灾事故以及财产损害。在电路设计时，请充分考虑安全性，采用合适的如冗余设计、利用非易燃材料以及故障或者事故防止等的安全设计方法。

关于利用本资料时的注意事项

1. 本资料是为了让用户根据用途选择合适的瑞萨科技产品的参考资料，不转让属于瑞萨科技或者第三者所有的知识产权和其它权利的许可。
2. 对于因使用本资料所记载的产品数据、图、表、程序、算法以及其它应用电路的例子而引起的损害或者对第三者的权力的侵犯，瑞萨科技不承担责任。
3. 本资料所记载的产品数据、图、表、程序、算法以及其它所有信息均为本资料发行时的信息，由于改进产品或者其它原因，本资料记载的信息可能变动，恕不另行通知。在购买本资料所记载的产品时，请预先向瑞萨科技或者经授权的瑞萨科技产品经销商确认最新信息。
   本资料所记载的信息可能存在技术不准确或者印刷错误。因这些错误而引起的损害、责任问题或者其它损失，瑞萨科技不承担责任。
   同时也请通过各种方式注意瑞萨科技公布的信息，包括瑞萨科技半导体网站。
   （http：//www.renesas.com）
4. 在使用本资料所记载部分或者全部数据、图、表、程序以及算法等信息时，在最终做出有关信息和产品是否适用的判断前，务必对作为整个系统的所有信息进行评价。由于本资料所记载的信息而引起的损害、责任问题或者其它损失，瑞萨科技不承担责任。
5. 瑞萨科技的半导体产品不是为在可能和人命相关的环境下使用的设备或者系统而设计和制造的产品。在研讨将本资料所记载的产品用于运输、交通车辆、医疗、航空宇宙用、原子能控制、海底中继器的设备或者系统等特殊用途时，请与瑞萨科技或者经授权的瑞萨产品经销商联系。
6. 未经瑞萨科技的书面许可，不得翻印或者复制全部或者部分资料的内容。
7. 如果本资料所记载的某产品或者技术内容受日本出口管理限制，必须在得到日本政府的有关部门许可后才能出口，并且不准进口到批准目的地国家以外的国家。
   禁止违反日本和（或者）目的地国家的出口管理法和法规的任何转卖、挪用或者再出口。
8. 如果需要了解本资料所记载的信息或者产品的详细，请与瑞萨科技联系。