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

使用 I/O 端口的 I²C™ EEPROM I/F 实现方法

内容

本应用说明叙述 I²C 总线接口的概要，并且对通过 2 条信号线将 SLP 系列 38024（I²C 主器件）连接到 Mircochip 公司产的 24AA16 16K I²C 串行 EEPROM（I²C 从属器件）的应用进行了详细说明。

要点

H8/300L Super Low Power（SLP）系列具有各种内部功能，简化了在设计嵌入式应用系统方面的系统设计。内部功能有低功耗模式、时钟振荡器、监视定时器（WDT）、串行通信接口（SCI3）和各种外部/内部中断等。

但是，根据应用还需要追加外部的外围功能（I/O 扩展和专用存储器等）。作为标准的外围总线，能使用称为 Inter-IC（I²C）总线的 2 条信号线的双向串行接口。SLP 系列没有内置专用的 I²C 硬件，但是能通过软件控制 2 个 I/O 管脚，模拟此串行总线接口。

动作确认器件

H8/300L Super Low Power（SLP）系列– H8/38024

目录

1. I²C™接口的概要

I²C 总线使用串行数据信号（SDA）和串行时钟信号（SCL）的 2 条信号线接口，进行连接在总线上的器件间的信息交换。总线上的各器件具有固有的地址，根据功能起着发送器或者接收器的作用。

另外，器件具有主器件和从属器件的功能。主器件是进行传送的开始、控制（生成所有的帧信号和时钟信号）和结束的器件，从属器件是由主器件存取的器件。

（I²C 的一般特点叙述在有关 SPI 和 I²C 的应用说明中）

本章说明以下 3 点：

1. I²C 的控制
2. 写操作
3. 读操作

1.1 I²C EEPROM 的控制

1.1.1 START（开始）和 STOP（停止）条件

I²C 总线通过总线主器件生成的 2 种不同的总线状态控制（帧形成）数据传送。当总线空闲时，2 条信号线都为高电平。2 种总线状态为开始（START）位条件和停止（STOP）位条件。

开始条件是 SDA 从高电平迁移到低电平且 SCL 为高电平的状态。停止条件是 SDA 从低电平迁移到高电平且 SCL 为高电平的状态。当 SCL 为高电平时，SDA 上的数据必须总是有效（稳定）；只有在 SCL 为低电平时，SDA 信号电平才发生变化。在 SCL 的每 1 个周期发送 1 位数据。

接着开始条件被发送的总线信息的起始字节（8 位）是 7 位从属地址字段和数据方向（R/$\overline{W}$）位（此说明仅以 I²C 的 7 位寻址方式为对象）。数据方向位（最低位）指定是主器件给从属器件发送（0 = 写）数据还是从从属器件接收（1 = 读）数据。

应答位是 SDA 为低电平信号，在主器件送来的应答时钟脉冲（字节发送的第 9 个高电平 SCL 时钟）期间由接收器件（主器件或者从属器件）发送。如果接收器不能接收数据（占线状态的从属接收器）或者要指示数据条件的结束（接收器是主器件），就发送否认应答（在第 9 个高电平 SCL 时钟期间将 SDA 置为高电平）。

在发送开始条件和从属地址后，继续根据主器件和接收器之间的要求进行数据传送。当传送了最后字节并且返回了应答时，主器件就立即发行停止条件，结束总线的使用。

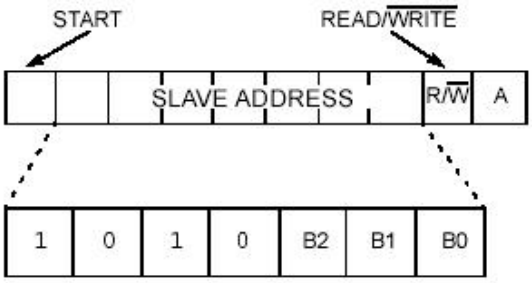
通过上述的 I²C 总线运行概要，可了解在使用串行 EEPROM 接口方面所需的基本事项（在 I²C 7 位寻址方式的实用方面所需的知识）。

（对于 I²C 的详细结构，请参照有关 SPI 和 I²C 的应用说明）

（另外，关于 I²C 总线支持的各模式的详细内容，请参照 Koninklijke Philips Electronics N.V.公司发行的 I²C 总线规格说明）

1.1.2 器件地址的指定

主器件送来的开始条件后的起始字节是控制字节。控制字节有 4 位控制码，在 24AA16 的读操作和写操作的情况下为 2 进制数 1010。



控制字节的后 3 位是块选择位（B2、B1、B0）。主器件通过此块选择位从 8 个 256 个字的存储块中选择存取块。这 3 位实际上是字地址的高 3 位，因为根据 I²C 协议定存储容量为 256 个字× 8 块，所以 1 个系统能支持的 24AA16 存储器只有 1 个，必须注意。

控制字节的最后 1 位定义应该执行的操作。当此位为“1”时是读操作；为“0”时是写操作。在开始条件之后，24AA16 监视 SDA 总线，检查被发送来的器件类型 ID。如果检测到代码 1010，就作为从属器件将应答信号输出到 SDA。24AA16 根据 R/$\overline{W}$ 位选择读或者写操作。

Operation	Control Code	Block Select	R/$\overline{W}$
Read	1010	Block Address	1
Write	1010	Block Address	0

【注】 控制字节因器件而不同。关于控制字节的详细内容，请在使用前参照器件的数据表或者 I²C 总线规格说明。

1.1.3 位传送和数据的有效性

在开始条件和停止条件之间，从发送器发送给接收器的数据字节数没有限制，能由主器件决定。各字节（固定 8 位）从最高位开始串行传送，最后为应答位。

开始条件之后，在时钟信号为高电平期间数据信号稳定。此时的信号为有效数据。

必须在时钟信号为低电平期间更改信号上的数据。数据的每 1 位，时钟为 1 个脉冲。

1.1.4 应答

被地址指定的接收器件必须在每接收 1 个字节时输出应答，主器件必须根据此应答位，追加输出 1 个脉冲的时钟。

在内部写周期中，24AA16 不输出应答位。

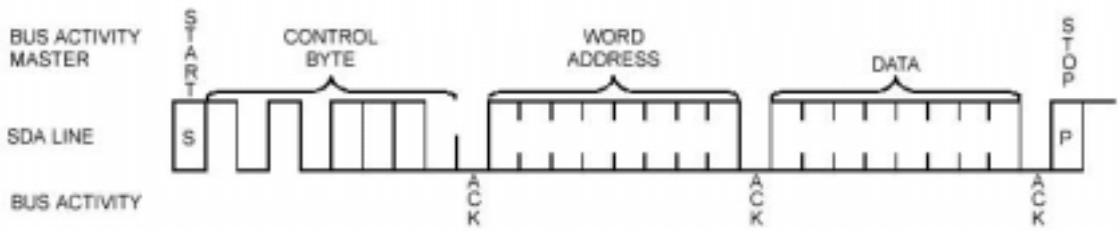
1.2 写操作

如果将从属地址的 R/$\overline{W}$ 位置“0”， 就开始写操作。写操作有字节写和页写 2 种。

1.2.1 字节写

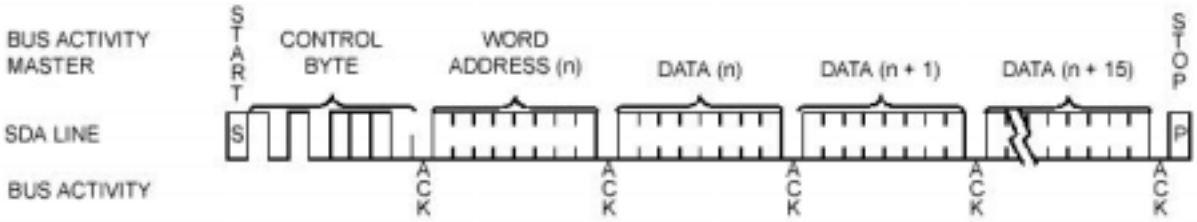
字节写操作能写随机的 EEPROM 地址，需要发送以下内容：

- 开始条件（主器件）
- EEPROM 器件地址和 R/$\overline{W}$ 位= 0（主器件）
- 应答位（EEPROM）
- 写对象的 EEPROM 字地址（主器件）*1
- 应答位（EEPROM）
- 被写的数据字节（主器件）
- 应答位（EEPROM）
- 停止条件（主器件）



1.2.2 页写

页写操作和上述的字节写操作大致相同，但是有一点不同，即字节写只写 1 个字节，而页写能在传送 EEPROM 地址和停止位期间传送 16 个字节*2。在页写操作中，EEPROM 每个字节自动递增内部地址指针。



【注】 1. 字地址的个数因器件而不同，详细内容请参照相关器件的数据表。
2. 页写操作的字节数因器件而不同，请确认相关器件的数据表。

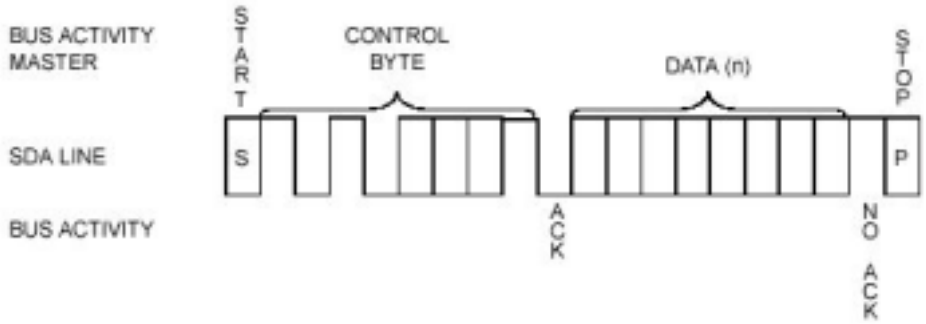
1.3 读操作

读操作支持当前地址读、随机读和顺序读 3 种。读操作用和写操作同样的方法开始，但是有一点不同，即器件地址字节的 R/$\overline{W}$ 位被置“1”。

1.3.1 当前地址读

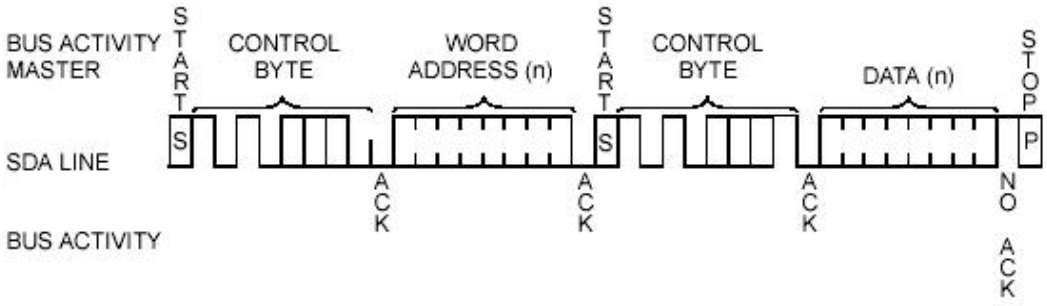
在当前地址读模式中，因为从属器件传送到主器件的数据是从前一次存取的地址+1 的位置读取的内容，所以不需写 EEPROM 字节地址。此类型的读操作如下：

- 开始条件（主器件）
- EEPROM 器件地址和 R/$\overline{W}$ 位= 1（主器件）
- 应答位（EEPROM）
- 被读的数据字节（是指定的从属器件从前一次存取的存储器地址+1 的位置传送的 EEPROM 字节）
- 否认应答位（主器件）
- 停止条件（主器件）



1.3.2 随机读

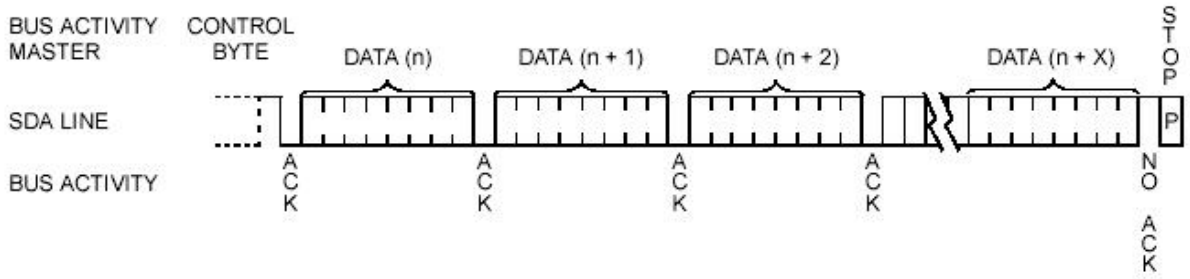
随机读模式以虚字节写周期开始（主器件以开始条件、器件地址和对象字地址*[1] 的顺序发送），然后是上述的当前地址模式周期。



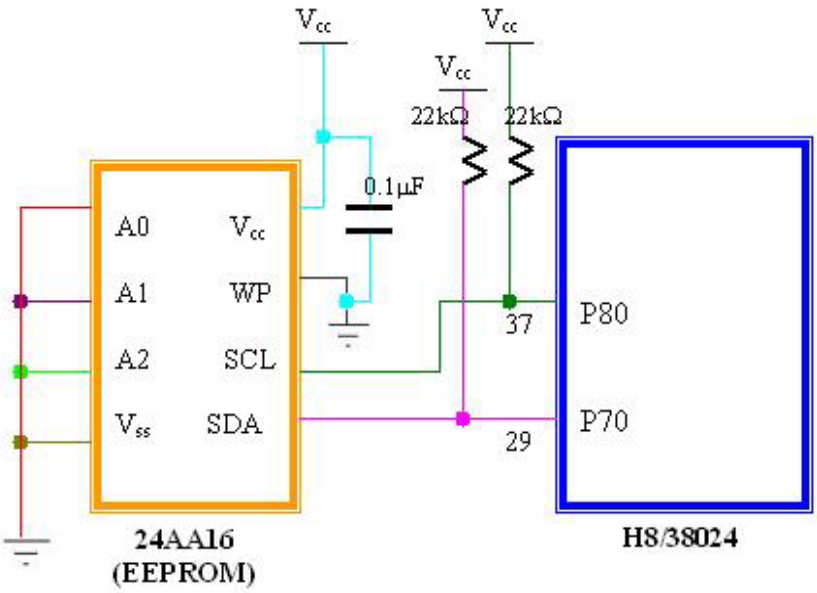
【注】 1. 字地址的个数因器件而不同，详细内容请参照相关器件的数据表。

1.3.3 顺序读

顺序读模式以随机读开始。主器件在传送 1 个字节后不以否认应答结束读操作，而是在取到各字节数据后返回有效的应答信号。从属 EEPROM 接受此应答后继续读操作，发送下一个字节数据。顺序读操作继续到主器件在读取最后字节后发行否认应答使之结束为止，最后发送停止条件。



2. 硬件设计



*Signal Name

A0,A1,A2	*User Configuration Chip Selects*
V_{ss}	*Ground*
SDA	*Serial Data*
SCL	*Serial Clock*
WP	*Write Protected Input*
V_{cc}	*Supply Voltage*

3. 函数的概要

在 I2C.c 中使用的函数：

- void main(void)

在 RW.c 中使用的函数：

- unsigned char SclIn (void)
- unsigned char SdaIn (void)
- void SclOut (unsigned char)
- void SdaOut (unsigned char)
- void Delay(void)
- void Delay2x(void)
- unsigned char CheckBusState(void)
- void SendStartBit(void)
- void SendBit (unsigned char)
- unsigned char GetBit (void)
- unsigned char GetAck (void)
- unsigned char SendByte(unsigned char)
- unsigned char GetByte(void)
- void SendStopBit(void)
- unsigned char I2cWrite(unsigned char, unsigned char *,unsigned char, unsigned char)
- unsigned char I2cRead(unsigned char, unsigned char *,unsigned char, unsigned char)
- char CheckWriteReady(void)
- unsigned char I2cCurrentRead(unsigned char , unsigned char *)

void main(void)

main 函数演示和测试 I²C 协议的 EEPROM 控制。通过字节写和页写进行 EEPROM 的写操作，然后通过当前地址读、顺序读或者随机读进行 EEPROM 的读操作。

unsigned char SclIn (void)

本函数进行是将端口管脚 SclIn 用作输入还是用作输出的控制。如果将位清 0，就为输入管脚。另外，检查 SCL 端口是低电平还是高电平。

unsigned char SdaIn (void)

本函数进行是将端口管脚 SdaIn 用作输入还是用作输出的控制。如果将位清 0，就为输入管脚。另外，检查 SDA 端口是低电平还是高电平。

void SclOut (unsigned char)

参数： SCL 信号电平的检查结果 (unsigned char)

本函数进行是将端口管脚 SclOut 作为输入还是用作输出的控制。如果将位置 1，就为输出管脚。

void SdaOut (unsigned char)

参数： SCL 信号电平的检查结果 (unsigned char)

本函数进行是将端口管脚 SdaOut 用作输入还是用作输出的控制。如果将位置 1，就为输出管脚。

void Delay(void)
为了不错误发送不需要的信号，生成对应 SCL 下降沿的未确定区的最小内部延迟时间。

void Delay2x(void)
基于函数 void Delay(void)，生成 2 倍的延迟时间。

unsigned char CheckBusState(void)
本函数检测 I²C 总线是空闲状态（SCL 和 SDA 都为高电平）还是占线状态。

void SendStartBit(void)
本函数发送开始条件。

void SendBit (unsigned char)

参数： 发送的数据位(unsigned char)

发送位格式的数据。

unsigned char GetBit (void)
接收以位格式输入的数据。

unsigned char GetAck (void)
本函数和 GetBit 的功能相同，但是因为主器件在检测是否返回 ACK（SDA 为低电平）之前，必须将 SDA 置为高电平，所以是更加需要注意的操作。

unsigned char SendByte(unsigned char)

参数： 发送的字节数据 (unsigned char)

从最高位（MSB）开始发送字节。

unsigned char GetByte(void)
从最高位（MSB）开始接收字节。

void SendStopBit(void)
发送停止条件，结束运行。

unsigned char I2cWrite(unsigned char, unsigned char *,unsigned char, unsigned char)

参数： 从属地址 (unsigned char), 保存数据的缓冲器地址 (unsigned char*), 写数据的字地址(unsigned char), 数据的长度 (unsigned char)

本函数按以下步骤进行能适用于字节写和页写的写操作：

(1) 检查总线状态
(2) 发送开始条件
(3) 发送控制码和从属地址
(4) 发送写数据的字地址
(5) 按照写数据的长度，将数据写到 EEPROM
(6) 停止条件.

unsigned char I2cRead(unsigned char, unsigned char *,unsigned char, unsigned char)

参数： 从属地址 (unsigned char), 保存数据的缓冲器地址 (unsigned char*), 读数据的字地址(unsigned char), 数据的长度 (unsigned char)

本函数按以下步骤进行能适用于随机读和顺序读的读操作。

(1) 检查总线状态
(2) 发送开始条件
(3) 发送虚的控制码和从属地址
(4) 发送读操作的字地址
(5) 将 SDA 信号置为高电平
(6) 开始位
(7) 发送控制码和从属地址
(8) 按照读数据的长度，从 EEPROM 读出数据
(9) 停止条件

char CheckWriteReady(void)

因为 24AA16 等 Microchip 公司产的器件在写周期中不发送应答，所以使用本函数确定周期的结束。如果周期结束，器件就返回 ACK，主器件就能进入下一个读或者写命令。

unsigned char I2cCurrentRead(unsigned char , unsigned char *)

参数： 从属地址 (unsigned char), 保存数据的缓冲器地址 (unsigned char*),

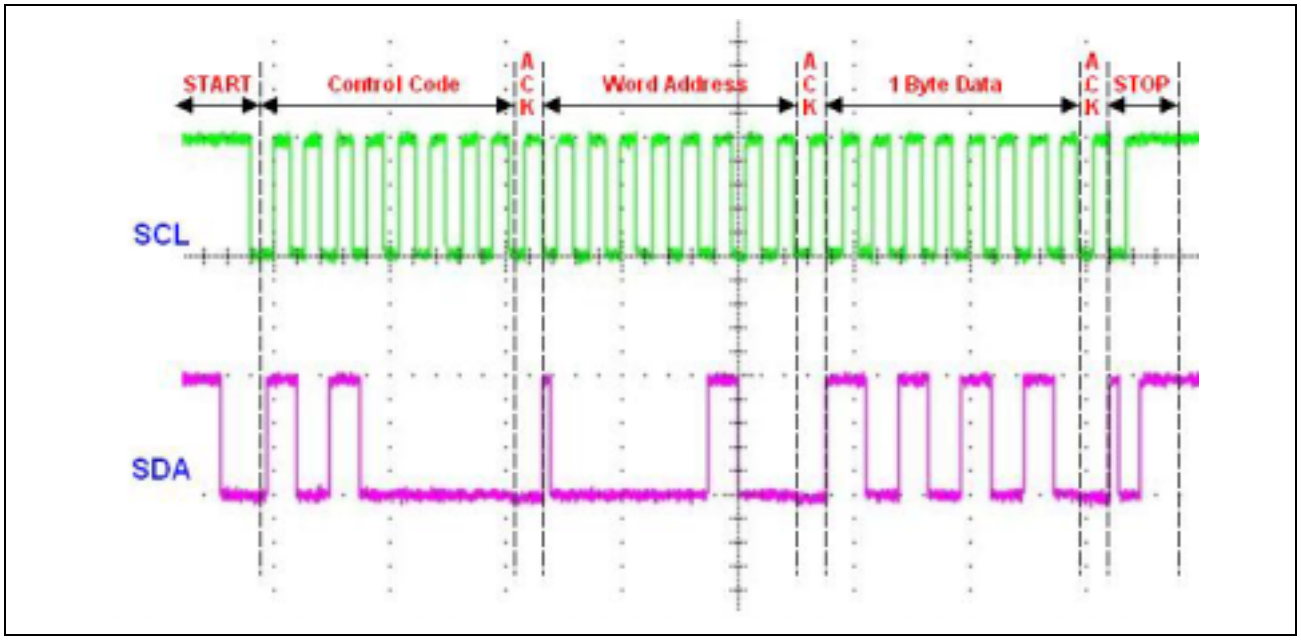
本函数按以下步骤进行能适用于随机读和顺序读的读操作：

(1) 检查总线状态
(2) 发送开始条件
(3) 发送控制码和从属地址
(4) 从 EEPROM 读取 1 个字节的数据
(5) 停止条件

4. 程序的解析

4.1 字节写

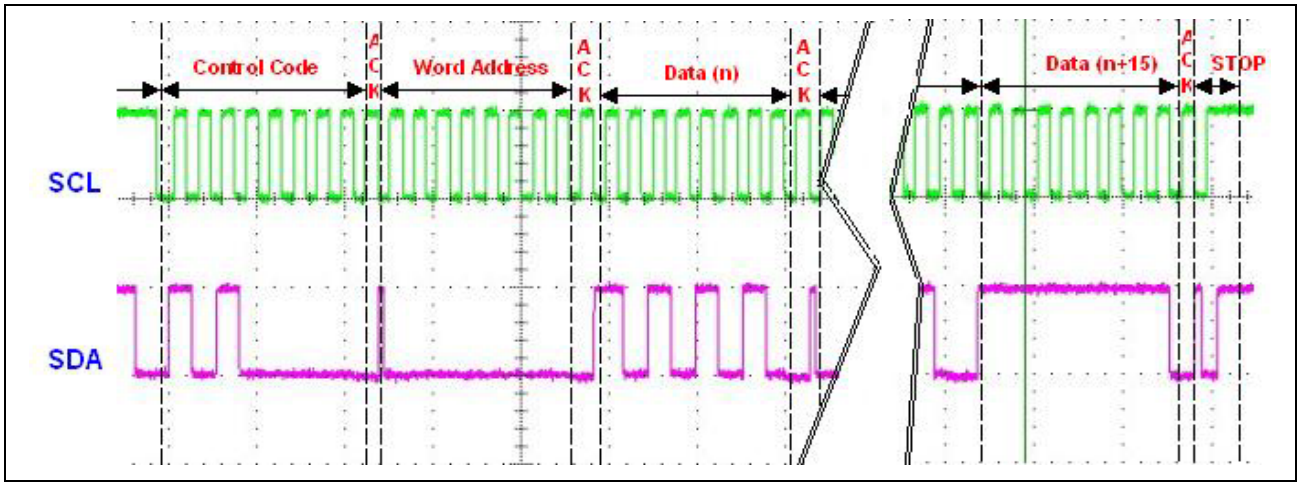
字节写操作如下图所示：



1 个字节的写操作时间平均为*16.0ms。

4.2 页写

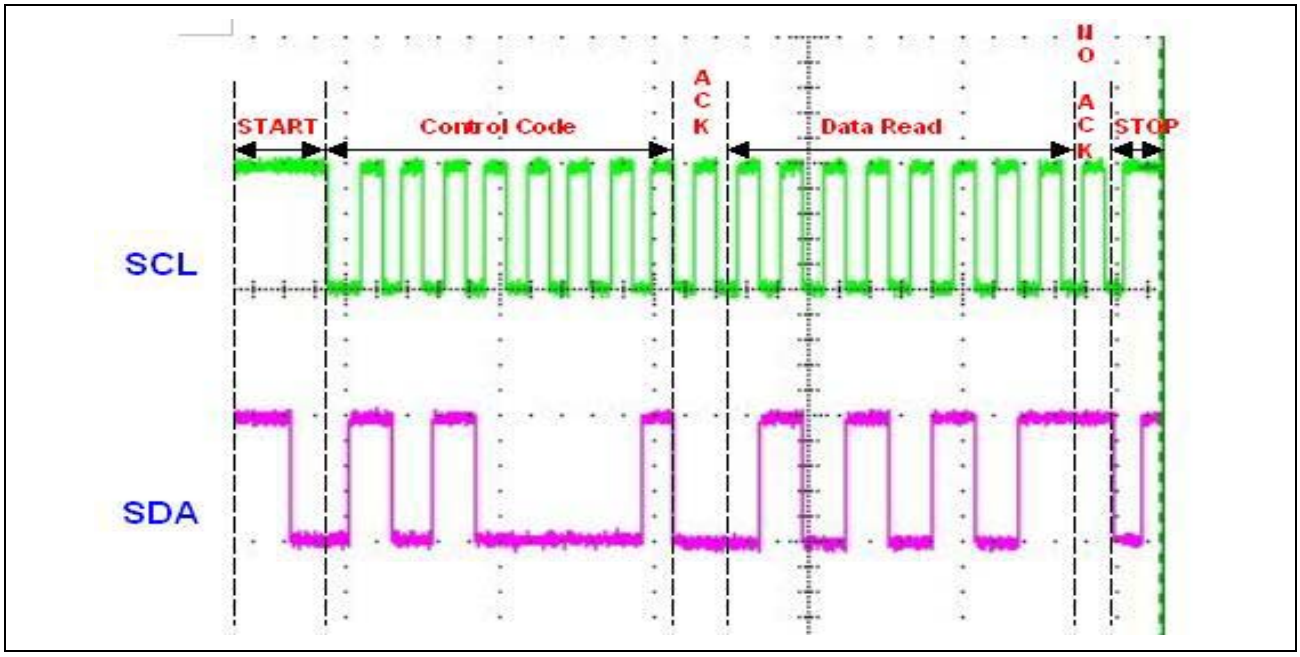
页写操作如下图所示：



1 页（16k 字节）的写操作时间平均为*89.5ms。

4.3 当前读

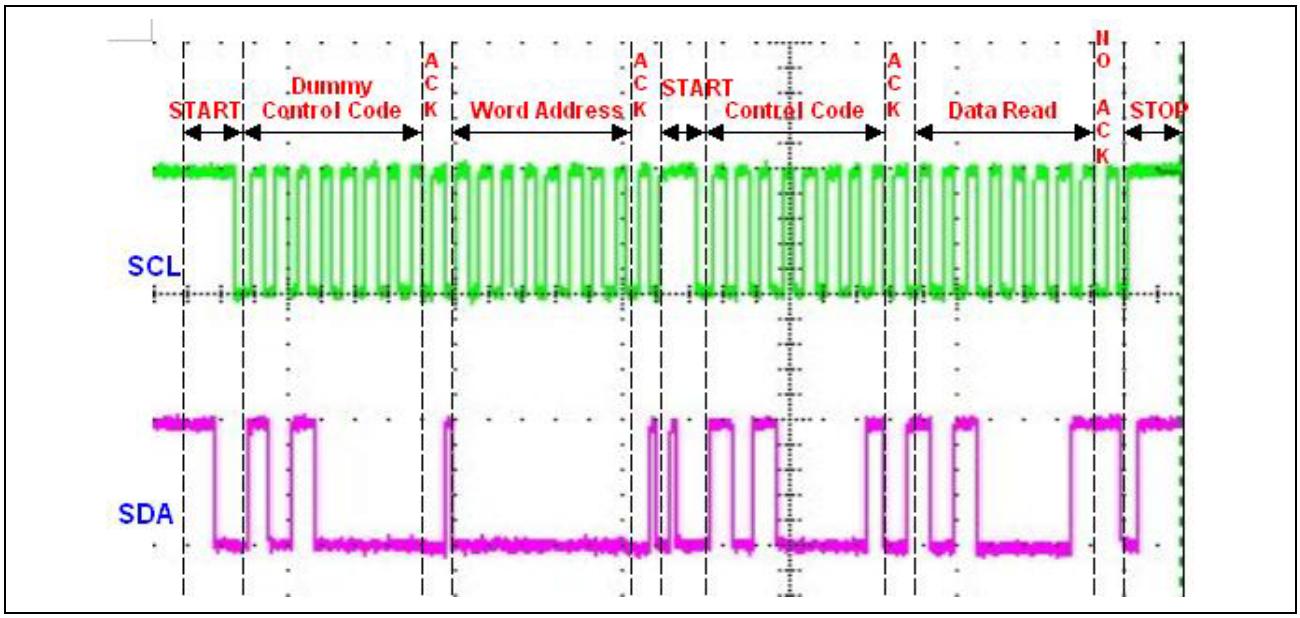
当前读操作如下图所示：



当前读的 1 个字节的读操作时间平均为*11.2ms。

4.4 随机读

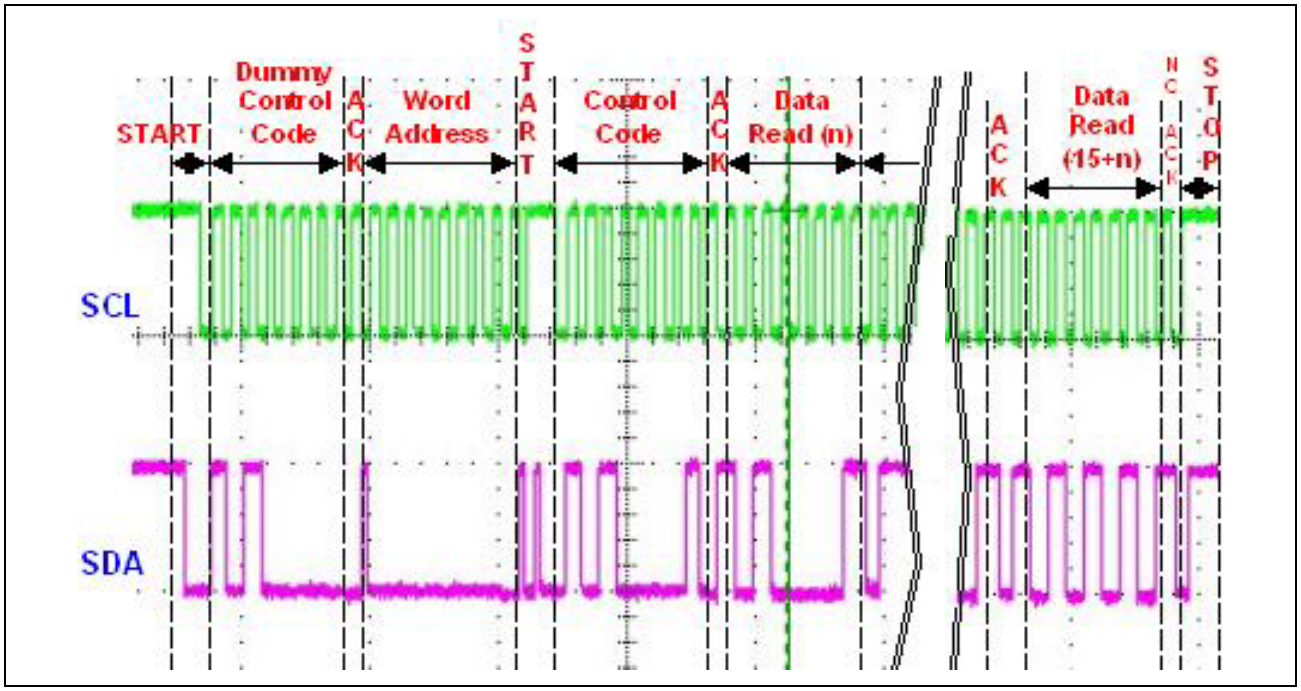
随机读操作如下图所示：



随机读的 1 个字节的读操作时间平均为*22.3ms。

4.5 顺序读

顺序读操作如下图所示：



顺序读的 1 页（16k 字节）的读操作时间平均为*97.8ms。

【注】* 上述的平均时间全部是在以下条件下的测量值：
(1) 10MHz 时钟
(2) 1/2 系统时钟分频比

5. 程序例

```
/************************************************************************/
/*                                                                      */
/*  FILE       :I2C.c                                                   */
/*  DATE       :Fri, Dec 27, 2002                                       */
/*  DESCRIPTION :Main Program                                           */
/*  CPU TYPE   :H8/38024F                                               */
/*                                                                      */
/*  This file is generated by Hitachi Project Generator (Ver.2.1).      */
/*                                                                      */
/************************************************************************/

#include "iodefine.h"
#include "I2C.h"
#include <stdio.h>

#define NODE_ADDR      0xa0

unsigned int i;

unsigned char buf[16]={0xaa, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77,
                       0x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff  };

void main(void)
{
  byte_write();

  for (i=0;i<1000;i++);             //delay approximately 500ms

  page_write();

  current_address_read();

  random_or_sequential_read();
}
```

```
void byte_write(void)
{
  //byte write
  i = I2cWrite(NODE_ADDR, buf, 1, 0x00);
     if (CheckWriteReady() ==1)
  i = I2cWrite(NODE_ADDR, buf, 2, 0x02);
     if (CheckWriteReady() ==1)
  i = I2cWrite(NODE_ADDR, buf, 1, 0x04);
}

void page_write(void)
{
  //page write
  i = I2cWrite(NODE_ADDR, buf, 16, 0x00);
}

void current_address_read(void)
{
  //current address read
  i = I2cCurrentRead(NODE_ADDR, buf);
}

void random_or_sequential_read(void)
{
  //random / sequential read
  i = I2cRead(NODE_ADDR, buf,16, 0x00);
}
```

```
/*****************************************************************/
/*                                                               */
/*  FILE       :RW.c                                             */
/*  DATE       :Fri, Dec 27, 2002                                */
/*  DESCRIPTION :Function Program                                */
/*  CPU TYPE   :H8/38024F                                        */
/*                                                               */
/*  This file is generated by Hitachi Project Generator (Ver.2.1).*/
/*                                                               */
/*****************************************************************/



#include "i2c.h"
#include "iodefine.h"

char CheckWriteReady(void);


/* Check SCL signal level */
unsigned char SclIn (void)
{
   SCL_IO_REG &= SCL_IO_RESET_BIT;        //It is Input
   if(SCL_DATA_REG & SCL_DATA_SET_BIT)    //check Port status
{
     return(HIGH);
   }
   else
   {
     return(LOW);
   }
}

/* Check SDA signal level */
unsigned char SdaIn (void)
{
   SDA_IO_REG &= SDA_IO_RESET_BIT;       //It is Input
   if(SDA_DATA_REG & SDA_DATA_SET_BIT)   //check Port status
{
      return(HIGH);
   }
   else
   {
     return(LOW);
   }
}
```

```
/* Drive SCL bus */
void SclOut (unsigned char status)
{
if (status == LOW)
{
    SCL_DATA_REG = 0;               //Drive Port LOW
    SCL_IO_REG |= SCL_IO_SET_BIT;   //Port is output

    }
    else
    {
        SCL_DATA_REG = 1;           //Port is Input & using external pull-up
                                    //resistor to go high
     SCL_IO_REG |= SCL_IO_SET_BIT;  //Port is output
      }
}

/* Drive SDA bus */
void SdaOut (unsigned char status)
{
    if (status == LOW)
    {
      SDA_DATA_REG = 0;               //Drive Port LOW
        SDA_IO_REG |= SDA_IO_SET_BIT;   //Port is output
    }
    else
    {
        SDA_DATA_REG = 1;             //Port is Input & using external pull-up
                                      //resistor to go high
    SDA_IO_REG |= SDA_IO_SET_BIT;    //Port is output
    }
}

void Delay(void)                      //Delay approximately 10ms
{
    unsigned char i;

    i=0;
    while(i<20)
    {
        i++;
    }
}

void Delay2x(void)                   //Delay approximately 20ms
{
    Delay();
    Delay();
}
```

```
/* All codes below here are independent with hardware, such as microprocessor,
I/O port */
unsigned char CheckBusState(void)
{
    if ((SclIn() == HIGH) && (SdaIn() == HIGH))
    {
      return(TRUE);
    }
    else
    {
      return(FALSE);
    }
}

/* It is nice to send out data at the middle of clock low */
void SendBit (unsigned char data_byte)
{
    SclOut(LOW);
    Delay();
    if (data_byte != 0)
    {
      SdaOut(HIGH);
    }
    else
    {
      SdaOut(LOW);
    }

    Delay();
    SclOut(HIGH);
    while (SclIn() != HIGH) {}  //wait for slow device to release clock
    Delay2x();
}

void SendStartBit(void)
{
    Delay();
    SdaOut(LOW);

    Delay2x();
    Delay2x();

    SclOut(LOW);
    Delay();
}
```

```
/* It is nice to sample data input at the middle of clock high */
unsigned char GetBit (void)
{
    unsigned char temp;

    SclOut(LOW);
    temp = SdaIn();
    Delay2x();

    SclOut(HIGH);
    while (SclIn() != HIGH) {}  //wait for slow device to release clock
    Delay();

    temp = SdaIn();
    Delay();

    return(temp);
}

/* Getting ACK is similar to GetBit, but it is a little tricky since master
must pull SDA high
before it find out whether there is ACK (SDA is low) or not */
unsigned char GetAck (void)
{
    unsigned char temp;

    SclOut(LOW);
    Delay();

    SdaOut(HIGH);
    temp = SdaIn();
    Delay();

    SclOut(HIGH);
    while (SclIn() != HIGH) {}  //wait for slow device to release clock
    Delay();

    temp = SdaIn();
    Delay();

    return(temp);
}
```

```
/* Note master need get ACK after sending out every byte */
unsigned char SendByte(unsigned char data_byte)
{
    unsigned char i;
    unsigned char mask;

    mask = 0x80;       //send out MSB first
    for (i=0; i<8; i++)
    {
       SendBit(data_byte & mask);
       mask /= 2;
     }

    return(GetAck());
}

unsigned char GetByte(void)
{
    unsigned char temp1, temp2;
    unsigned char i,mask;

    mask = 0x80;
    temp2 = 0;
    for (i=0; i<8; i++)
    {
       temp1 = GetBit() * mask;
       temp2 += temp1;
       mask /= 2;
    }

    return(temp2);
}

void SendStopBit(void)
{
    SclOut(LOW);
    Delay();

    SdaOut(LOW);
    Delay();

    SclOut(HIGH);
    Delay2x();

    SdaOut(HIGH);
}
```

```
unsigned char I2cWrite(unsigned char slave_addr, unsigned char *buf_ptr,
unsigned char length,unsigned char word_addr) {
    unsigned int i;

    if( CheckBusState() != TRUE)
    {
     return(BUS_BUSY);
    }

    SendStartBit();

    if (SendByte((slave_addr) & 0xfe) != LOW) //Send address and write command
    {
     return(NO_RESPONSE);
    }

    if (SendByte(word_addr) != LOW)           //Send low word address
    {
     return(NO_RESPONSE);
    }


   for(i=0; i<length; i++)
   {
       if(SendByte(*buf_ptr++) != LOW)
       {
           return(ERR_RESPONSE);
       }
   }

    SendStopBit();

    return(OP_DONE);
}
```

```
unsigned char I2cRead(unsigned char slave_addr, unsigned char *buf_ptr,
unsigned char length,unsigned char word_addr)
{
    unsigned char i=0,j=0;
    unsigned char DataBuffer[500];
    unsigned char LastData;

    if( CheckBusState() != TRUE)
    {
       return(BUS_BUSY);
     }

    SendStartBit();

    if (SendByte((slave_addr) & 0xfe) != LOW)  //Send address and read command
    {
       return(NO_RESPONSE);
     }

    if (SendByte(word_addr) != LOW)            //Send high word address
    {
     return(NO_RESPONSE);
     }

    SendStopBit();                              //Pull-up SDA line
    SendStartBit();

    if (SendByte((slave_addr) | 0x01) != LOW)  //Send address and read command
    {
       return(NO_RESPONSE);
     }

    for(i=0; i<length-1; i++)
    {
     DataBuffer[i]= GetByte();
     SendBit(LOW);                          //ack it low
     }

    DataBuffer[i]= GetByte();             //get last data and ack high

    SendBit(HIGH);
    SendStopBit();

    return(OP_DONE);
}
```

```
/*Since Microchip devices such as 24AA16 will not acknowledge during a write
cycle, this can be used to determined when the cycle is complete. So that the
master can proceed with next operation*/
char CheckWriteReady(void)
{
    unsigned int i=0;

    while(i<4)
    {
        SendStartBit();

        if (SendByte((0xa0) | 0x00) == LOW)
        {
         SendStopBit();
         return (1);
         }

        SendStopBit();
      i++;
      }
}

unsigned char I2cCurrentRead(unsigned char slave_addr, unsigned char *buf_ptr)
{
    unsigned char i=0,j=0;
    unsigned char DataBuffer[500];
    unsigned char LastData;

    SendStartBit();

    if (SendByte((slave_addr) | 0x01) != LOW)  //Send address and read command
    {
        return(NO_RESPONSE);
      }

    *buf_ptr = GetByte();                    //get last data and ack high

    SendBit(HIGH);
    SendStopBit();

    return(OP_DONE);
}
```

```
/*****************************************************************/
/*                                                               */
/*  FILE       :i2c.h                                            */
/*  DATE       :Fri, Dec 27, 2002                                */
/*  DESCRIPTION :Definition of constant and functions            */
/*  CPU TYPE   :H8/38024F                                        */
/*                                                               */
/*  This file is generated by Hitachi Project Generator (Ver.2.1).*/
/*                                                               */
/*****************************************************************/

//bit patterns for EEPROM access instructions
#define READSR 0xA0 /* (bit reversed 05) */
#define SETWEL 0x60 /* (bit reversed 06) */
#define WRITE 0x40  /* (bit reversed 02) */
#define READ 0xC0   /* (bit reversed 03) */

/* Set as 1010 binary for read and write operation (device dependent)*/
#define NODE_ADDR      0xa0

/* Misc */
#define HIGH    1
#define LOW     0

#if !defined(TRUE)
#define TRUE    1
#endif

#if !defined(FALSE)
#define FALSE   0
#endif

#define OP_DONE         0x00
#define BUS_BUSY        0x01
#define NO_RESPONSE     0x02
#define ERR_RESPONSE    0x04

// SDA and SCL port def.

/* control SDA port as input or output */
#define SDA_IO_REG             P_IO.PCR7.BYTE
#define SDA_IO_SET_BIT         0x01    //output
#define SDA_IO_RESET_BIT       0xfe    //input

/* check SDA port low or high */
#define SDA_DATA_REG           P_IO.PDR7.BYTE
#define SDA_DATA_SET_BIT       0x01
#define SDA_DATA_RESET_BIT     0xfe

/* control SCL port as input or output */
#define SCL_IO_REG             P_IO.PCR8.BYTE
#define SCL_IO_SET_BIT         0x01    //output
#define SCL_IO_RESET_BIT       0xfe    //input
```

```
/* check SCL port low or high */
#define SCL_DATA_REG           P_IO.PDR8.BYTE
#define SCL_DATA_SET_BIT       0x01
#define SCL_DATA_RESET_BIT     0xfe

//I2C modules
void byte_write(void);
void page_write(void);
void current_address_read(void);
void random_or_sequential_read(void);
void startbit(void);
void stopbit(void);
void sendbit(unsigned char );
void controlbyte(unsigned char );
void ReadBuffer(void);
extern char checkwrirdy();
unsigned char getACK(void);
unsigned char getbit(unsigned char);
unsigned char I2cWrite(unsigned char slave_addr, unsigned char *buf_ptr,
unsigned char length,unsigned char word_addr);
unsigned char I2cRead(unsigned char slave_addr, unsigned char *buf_ptr,
unsigned char length,unsigned char word_addr);
unsigned char I2cCurrentRead(unsigned char slave_addr, unsigned char
*buf_ptr);
```

6. 参考文献

1. The I²C-Bus Specification (Version 2.1), January 2000, Koninklijke Philips Electronics N.V.
2. 24AA16/ 24LC16B 16K I²C Serial EEPROM, 2002, Microchip Technology Inc.
   http://www.microchip.com/download/lit/pline/memory/ic/21703b.pdf
3. http://www.esacademy.com/faq/i2c/
4. H8/38024 Series, H8/38024F-ZTAT Hardware Manual (version 2.0), 20 Feb 2002, Hitachi Ltd.
5. Leonard Haile, Hitachi H8/3437 Series Microcontroller I²C Peripheral-A practical SMBus/I²C Firmware Design Guide (Revision 1.2), 12 June 1998, Hitachi Semiconductor (America) Inc.

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。