To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

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

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

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

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

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

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

Application Note

32

SH7144F Group On-Chip Interface I^{2}C Bus Interface Volume

Application Note

Renesas 32-Bit RISC Microcomputer SuperH™ RISC engine Family/ SH7144 Series

Renesas Electronics
www.renesas.com

Rev.1.00 2003.7

Renesas 32-Bit RISC Microcomputer
SuperH™ RISC engine Family/
SH7144 Series

SH7144F Group
On-Chip Interface
I^2C Bus Interface Volume

Application Note

RENESAS

REJ05B0080-0100O

Cautions

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.

Preface

A process of unification and standardization of peripheral interfaces is currently underway in various fields in response to the need for lower cost and greater ease of use. The I^2C Bus interface covered by this Application Note is one such standardized interface, and its applications include use as a consumer product control IC interface, notebook PC battery pack control interface, and PC monitor control interface.

The I^2C Bus is a bidirectional serial bus system standard developed by Philips Corporation of the Netherlands. With products conforming to this specification, mutual data communication is possible among a number of peripheral ICs using two lines (a clock line and data line).

The I^2C Bus interface incorporated in Renesas Technology's SH7144F Group of original 32-bit single-chip microcomputers conforms to the I^2C Bus interface proposed by Philips Corporation, and is provided with subset functions. (Note, however, that some I^2C Bus interface specifications may not be met.)

In this Application Note, section 1 and section 2 give an overview of the I^2C Bus and a general description of the specifications and functions of Renesas Technology's I^2C Bus interface module, and section 2 presents some examples of I^2C bus interface applications using the SH7144F Group.

Although the operation of the task programs in this Application Note has been checked, operation should be confirmed again before any of these programs are actually used.

Note: I^2C Bus: Inter IC Bus

Contents

Section 1 I^2C Bus Overview

1.1 I^2C Bus Features

1.1.1 I^2C Bus Features

I^2C Bus Features are summarized below.

- The bus comprises two bus lines: a serial data line (SDA) and serial clock line (SCL). I^2C Bus device expansion can be carried out easily.
- There is always a master-slave relationship between devices, and each device has a unique address. A device functioning as a master first specifies the unique address of the communicating party, thereby establishing a communication path and enabling data communication.
- Any device can become a master (and a multi-master system can be constructed). Therefore, with the I^2C Bus interface, a bus right contention prevention system is defined to prevent destruction of data.
- The data transfer speed is a maximum of 100 kbps in standard mode, and a maximum of 400 kbps in high-speed mode (a speed of up to 3.4 Mbps is defined in I^2C Bus Specification Ver. 2.0).
- The total number of devices in an I^2C Bus system is determined by the system capacitive load upper limit of 400 pF.
- Application examples include SMBus*1 and ACCESS.bus*2.

Notes: *1 SMBus (System Management Bus) is a serial bus developed by Duracell Corporation and Intel Corporation.

*2 ACCESS.bus is a serial bus developed by Digital Equipment Corporation.

1.1.2 Differences from Serial Interface (SCI)

Differences from Renesas Technology's serial interface, Serial Communication Interface (SCI), are summarized below.

As shown in table 1.1, the SCI uses two data lines, a transmission data line and a reception data line. Data communication is usually carried out on a one-to-one basis. The I^2C Bus, on the other hand, performs bidirectional communication using a single data line. As the destination communicating party is determined by having the master device specify the unique address of the communicating party, data transmission/reception is possible to and from a number of arbitrary devices. Also, since a bus right collision prevention mechanism is defined for the I^2C Bus, it supports a multi-master system in which any device can operate as a master. The transfer rate is a maximum of 100 kbps in standard mode and a maximum of 400 kbps in high-speed mode.

Table 1.1 Differences from SCI

	SCI		I^2C Bus
	Synchronous Communication	**Asynchronous Communication**	
Pins used	3-line system	2-line system	2-line system
	Transmit data output	Transmit data output	Transmit/receive data input/output
	Receive data input	Receive data input	
	Serial clock	Serial clock (when using external clock)	Serial clock
Transfer rate	100 bps to 4 Mbps	100 bps to 4 Mbps	100 kbps (standard mode) 400 kbps (high-speed mode)
Transmission/ reception to/from multiple ICs	No	No	Yes, slave controlled by address

Note: Hs mode (maximum transfer speed: 3.4 Mbps) defined in I^2C Bus Specification Ver. 2.0 is not supported.

1.1.3 I²C Bus Connection

Figure 1.1 illustrates I²C Bus interface connection. As shown in the figure, the I²C Bus comprises a clock line SCL and data line SDA, each of which is connected to bus power supply VBB via a pull-up resistance. The SCL pin and SDA pin of device 1 and device 2 are wired-AND connected to the SCL line and SDA line, respectively.

When device 1 drives the SCL line low, device 2 confirms that another device is using the bus by monitoring the state of the SCL line. Through the wired-AND connection, even if the SCL line is driven while device 1 is using the bus, it is possible for device 2 to drive SCL low and place the communication operation in a "waiting" state for device 1 (see section 2, SH7144F Group Application Examples, for details).

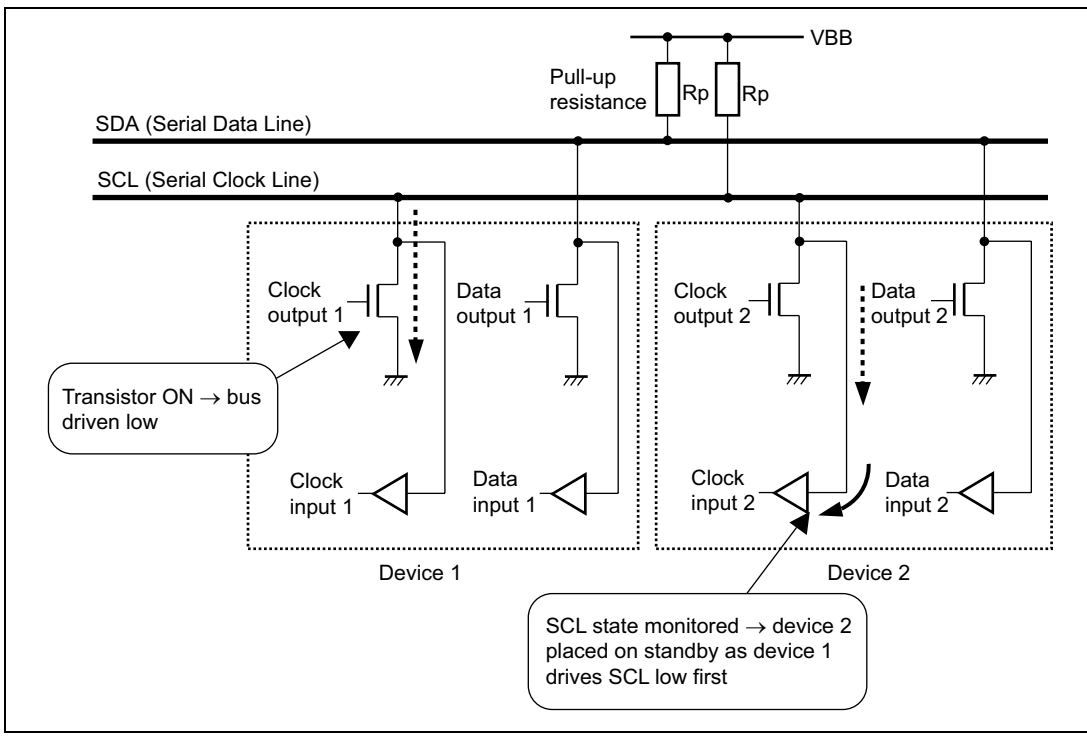


Figure 1.1 Bus Interface Connection (when Device 1 First Drives SCL Low)

1.2 Data Transfer Method Using I²C Bus

1.2.1 Basics of Data Transfer Using I²C Bus

First, the basics of data transfer using the I²C Bus will be explained.

(1) Master device

A master device generates a synchronization clock for performing data communication, and issues start/stop conditions for starting/stopping data communication.

(2) Slave device

A slave device is an I²C Bus device other than a master device. Its address is specified by a master device.

(3) Transmitting device

A transmitting device is a device that transmits data to the bus. It may be a master device or slave device.

(4) Receiving device

A receiving device is a device that receives data from the bus. It may be a master device or slave device.

(5) Start condition and stop condition

The start condition is an operation in which the SDA line changes from high to low when the SCL line is high. This starts data communication operation.

The stop condition is an operation in which the SDA line changes from low to high when the SCL line is high. This stops data communication operation.

The start condition and stop condition are always generated by a master. After a start condition has occurred, the bus goes to the busy state. When a stop condition occurs, the bus returns to the free state a short while later.

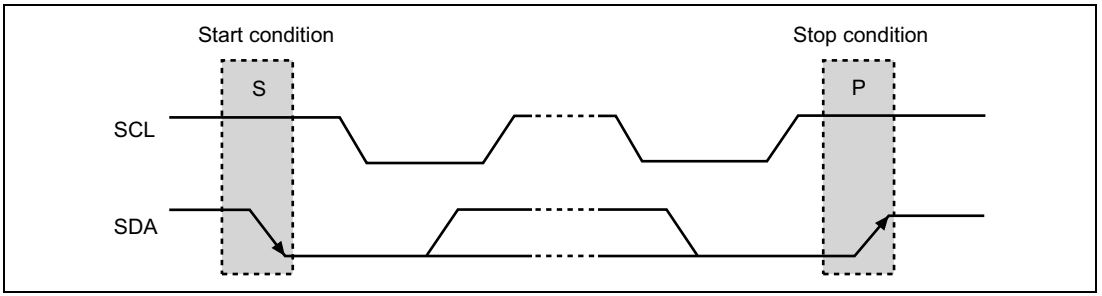


Figure 1.2 Start Condition and Stop Condition

(6) Data output timing

As data output timing, data on the SDA line is updated when the SCL line is low, and data on the SDA line is confirmed when the SCL line is high, as shown in figure 1.3. When the SCL line is high, the SDA line changes only in the event of a start condition or stop condition described above.

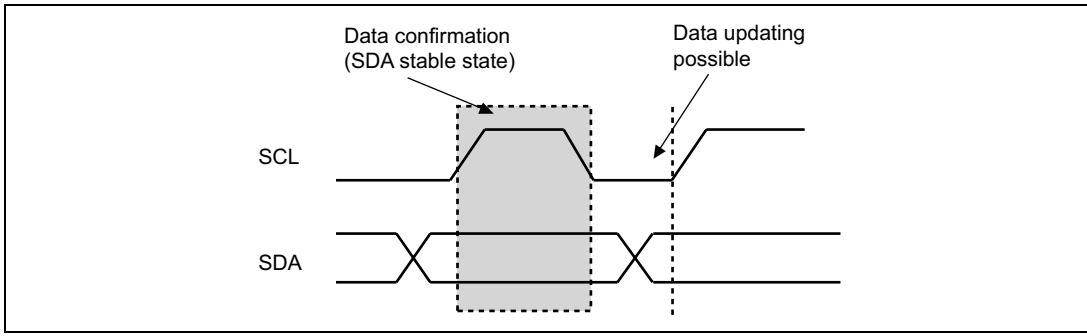


Figure 1.3 Data Output Timing

(7) Master transmission operation

A master transmission operation is an operation when a master device is a transmitting device. Possible master transmission operations are slave address transmission after issuance of a start condition, and transmission of a command, etc., to a slave device.

(8) Master reception operation

A master reception operation is an operation when a master device is a receiving device.

(9) Slave transmission operation

A slave transmission operation is an operation when a slave device is a transmitting device.

(10) Slave reception operation

A slave reception operation is an operation when a slave device is a receiving device. With a slave address transmit frame by a master device after a start condition, the slave device performs a reception operation.

(11) Bus released state

In this state, no I^2C Bus devices are performing data communication. The SCL and SDA lines are constantly high.

(12) Bus occupied state

In the bus occupied state, an I^2C Bus device is performing data communication. The bus released state is returned to when a master device issues a stop condition.

(13) Data transfer format

Figure 1.4 shows the I²C Bus data transfer format. Start and stop conditions, and the SCL clock, are generated by a master device. The first data after a start condition is a slave address, and a bit indicating the data communication direction is added as an 8th bit. A value of 0 in this bit indicates that data communication from the second byte onward is a master transmission operation, and a value of 1 means that data communication from the second byte onward is a master reception operation. The slave address is defined by 7 bits*1, and is set by the user in the range B'0000000 to B'1111111. However, address B'0000000 (known as the general call address*2) and some other addresses are reserved.

Transfer data comprises 1-byte (8-bit) units, and a confirmatory response bit (acknowledge bit) from the receiving device is added as the 9th bit. For example, when a master transmits a slave address, the corresponding slave drives SDA low at the 9th clock and returns an acknowledgment to the master. There is no restriction on the number of bytes of data that can be transferred between the first start condition and stop condition. Data communication is ended by a stop condition.

Notes: *1 In the I²C Bus Specification, a 10-bit address specification is defined. The Renesas Technology I²C Bus interface module does not support a 10-bit address specification.

*2 General call address B'0000000 is used to specify the addresses of all slaves connected to the bus.

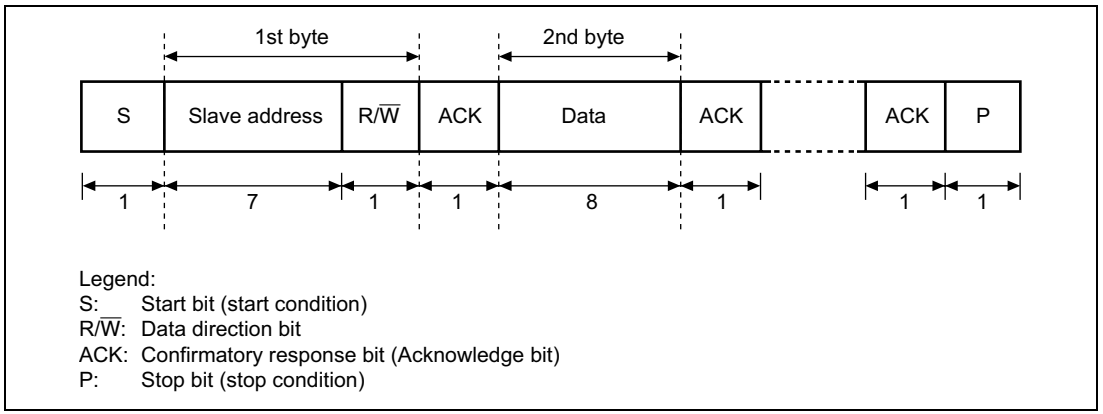


Figure 1.4 Data Transfer Format

1.2.2 Data Transfer Procedure

(Example: Master device = transmitting device, slave device = receiving device)

Figure 1.5 shows an example of a case in which a master device transmits one byte of data to a slave device. First, the master device issues a start condition, and when the SCL line is high, changes the SDA line from high to low. Next, the master outputs a clock on the SCL line, and also outputs the address of the slave to be communicated with on the SDA line. The slave address is defined by 7 bits, and a bit indicating the data communication direction is added as an 8th bit.

The master device releases the SDA line at the 9th clock, and prepares for an acknowledgment from the slave device. The slave device drives the SDA line low at the 9th clock and returns an acknowledgment. The master device receives the acknowledgment from the slave device, and holds the SCL line low until the next transmit data is ready. When the transmit data is ready, the master device outputs data to the SDA line while outputting a clock to the SCL line. As before, the slave device returns an acknowledgment to the master device at the 9th clock, reporting that data has been received normally. On receiving the acknowledgment from the slave device, the master device holds the SCL line low. The master device then issues a stop condition, and when the SCL line is high, the SDA line is changed from low to high.

If, during communication, the slave device is unable to receive data immediately because it is performing other processing, the SCL line can be held low on the slave device side, placing the master device in a waiting state. The timing at which the slave device can drive SCL low is when the master device is driving SCL low.

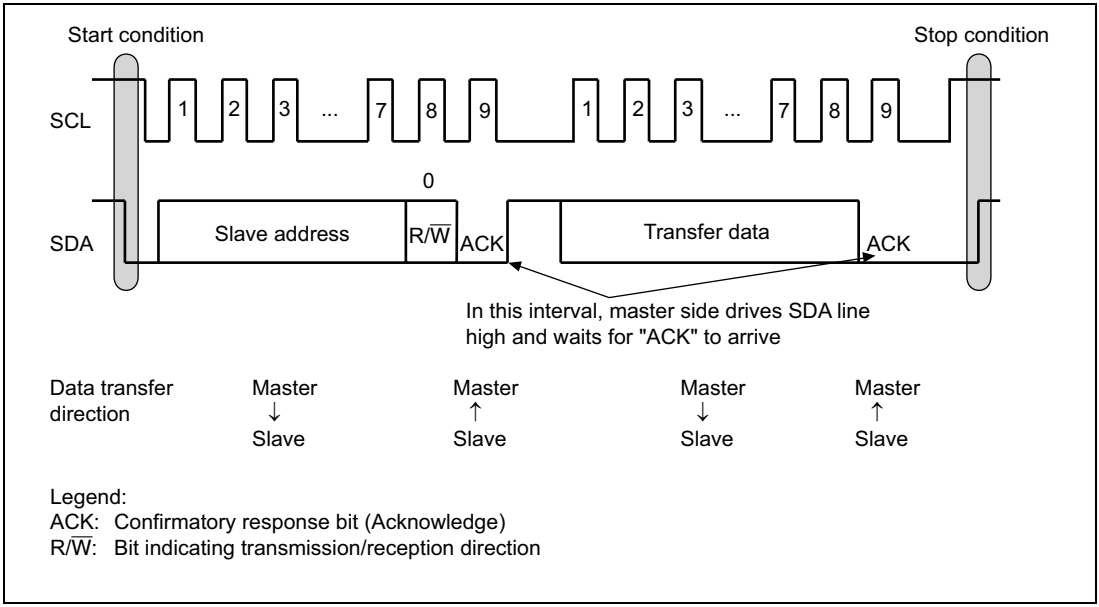


Figure 1.5 Data Transfer Format
(when Master = Transmitting Device, Slave = Receiving Device)

1.3 Single-Master and Multi-Master Configurations

1.3.1 Single-Master Configuration

A master device issues start and stop conditions, and manages data communication. It outputs a synchronization clock and slave address for data transmission/reception onto the SCL line. A system configuration such as that shown in figure 1.6, in which the master device does not change, is called a single-master configuration.

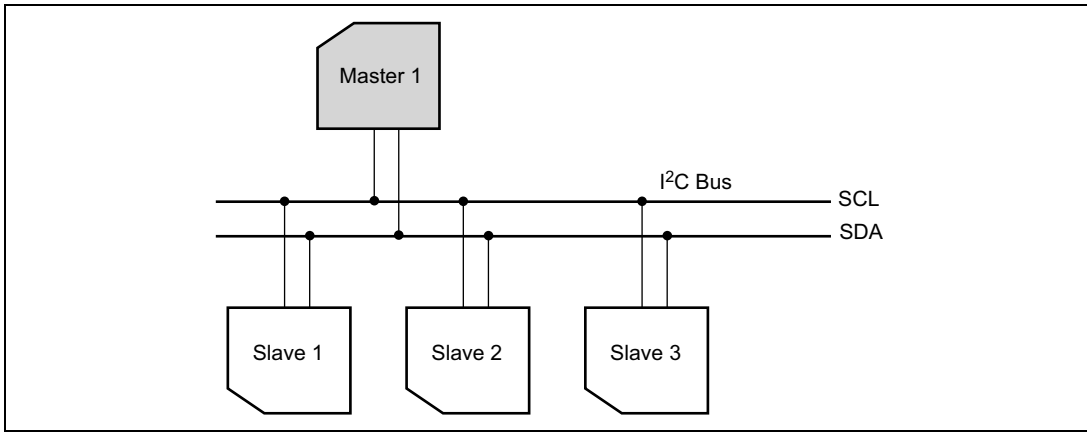


Figure 1.6 Single-Master Configuration

1.3.2 Multi-Master Configuration

A system configuration such as that shown in figure 1.7, in which there are two or more devices that can become a master device, is called a multi-master configuration.

A master device can start data transfer only when the bus is in the released state. In a multi-master configuration, it is possible that a number of master devices will attempt to start data transfer simultaneously. That is to say, bus right collisions will occur. Therefore, the I^2C Bus Specification stipulates a communication regulation procedure to be followed when a bus right collision occurs. See section 1.4, Communication Regulation Procedure, for details.

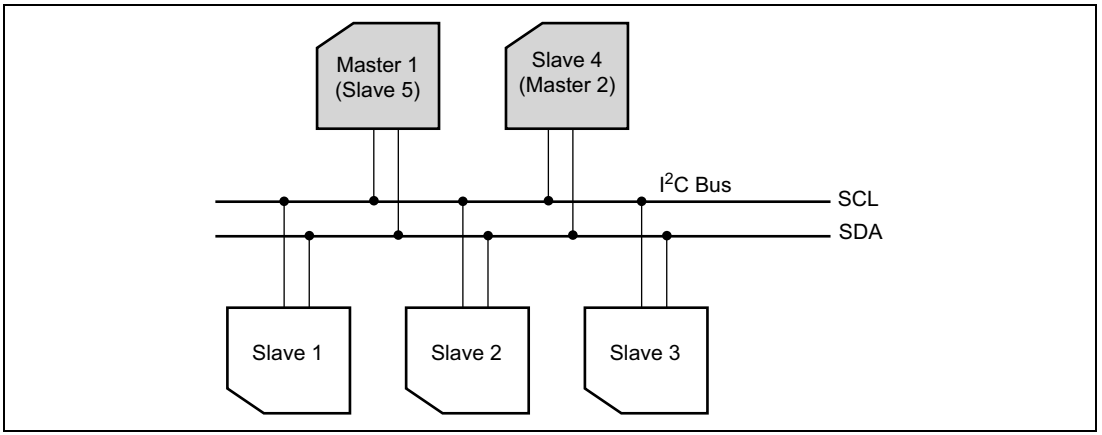


Figure 1.7 Multi-Master Configuration

1.4 Communication Regulation Procedure

With the I^2C Bus interface, a communication regulation procedure for preventing bus right collisions is defined, and can be applied to a multi-master configuration system.

A master device monitors the bus lines, confirms that the bus has been released, and issues a start condition. At this time, there is a possibility of start condition issuance processing being generated by a number of master devices simultaneously. A single master device is therefore determined by means of a communication regulation procedure as shown in figure 1.8.

With the I^2C Bus, data on the SDA line must be confirmed while the SCL line is high. Thus, each device monitors the SCL line for a rise after a start condition, and compares the state of the SDA line with the device's internal data (slave address). If device 1 drives SDA high and device 2 drives SDA low, the actual SDA line goes low due to the wired-AND connection, and therefore device 1 confirms a difference from the data it is trying to output, and turns off its data output stage. In this example, device 2 continues operation as the master device. If all the masters attempt to specify the address of the same slave device, the procedure moves on to the next step, and data comparison is carried out.

For example, if the transfer data is H'01 and H'02, as in figure 1.9, the interval during which data H'01 is low is longer, and so data H'01 is valid. Therefore, the general call address (H'00) has the highest priority.

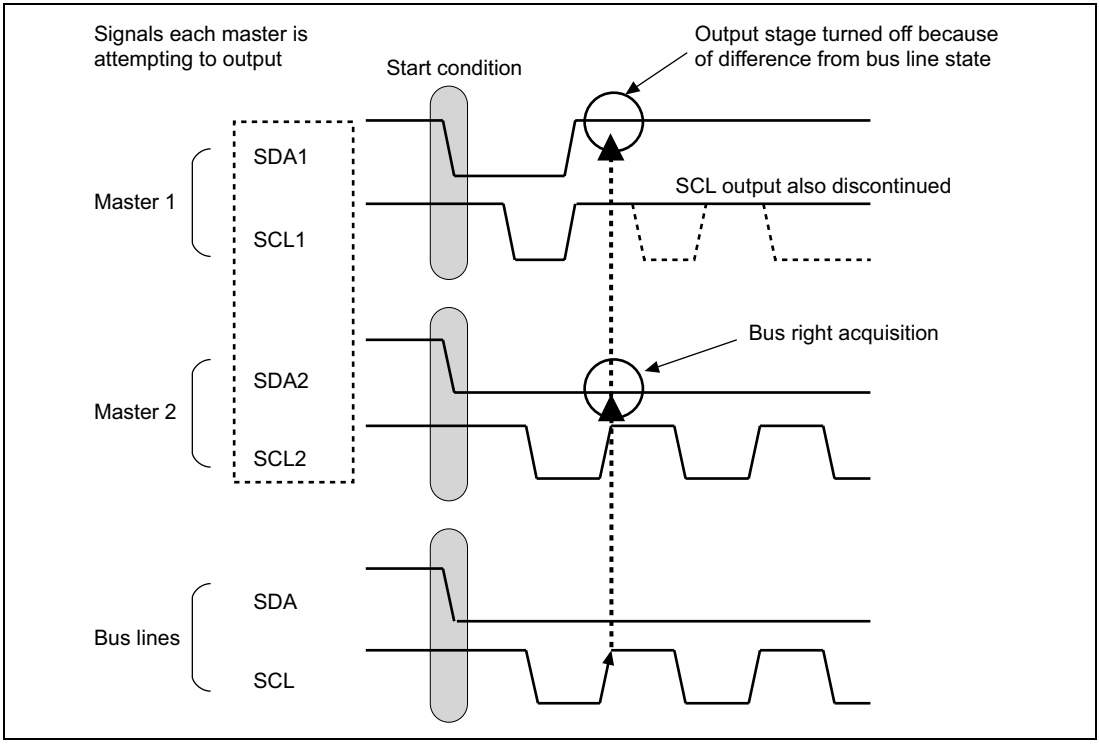


Figure 1.8 Communication Regulation Procedure (Detection of "Bus Arbitration Lost" State)

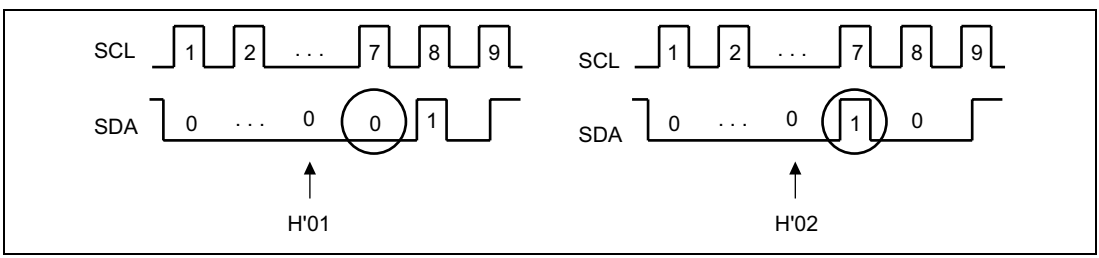


Figure 1.9 Actual Example of Communication Regulation

Section 2 SH7144F Group Application Examples

2.1 Guide to SH7144F Group Application Examples

2.1.1 Organization of SH7144F Group Application Examples

In these SH7144F Group application examples, the layout shown in figure 2.1 is employed to describe examples of the use of the SH7144F Group's I^2C Bus interface. The device used is assumed to be an SH7145F.

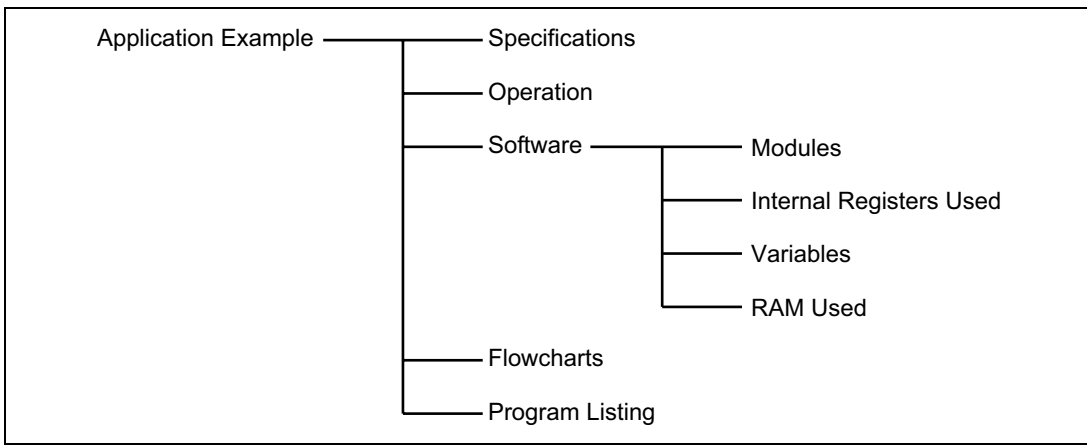


Figure 2.1 Organization of SH7144F Group Application Examples

(1) Specifications

Describes the system specifications for the sample task.

(2) Operation

Describes the operation of the sample task, using a timing chart.

(3) Software

(a) Modules

Describes the software modules used in the operation of the sample task.

(b) Internal Registers Used

Describes I^2C Bus interface and other internal registers set by the modules.

(c) Variables

Describes software variables used in operation of the sample task.

(d) RAM Used

Describes the labels and functions of RAM used by the modules.

(4) Flowcharts

Describes the software that executes the sample task, using flowcharts.

(5) Program Listing

Shows a program listing of the software that executes the sample task.

2.1.2 Vector Table Definition File

A vector table definition file using C is shown below. A file is created that secures the start addresses of interrupt service routines as shown in figure 2.2. When interrupt handling is used, the start label of the interrupt service routine is written at the vector location corresponding to the interrupt.

Figure 2.2 shows an example in which the main() function is used. The main() function and dummy() function are externally referenced.

The main() function is located at the power-on reset and manual reset vector address. The stack area at this time is allocated to on-chip RAM H'FFFFFFFC.

Dummy function dummy() is allocated for other interrupts.

```
/************************************************************/
/*    Filename   :   vector.c                                */
/*    Written    :   2003/2/1  REV.2.1                       */
/*    Purpose    :   SH7145F vector table                    */
/************************************************************/

/*------------ External Function Defnition -------------------------------*/
extern void main(void);             /* main function */
extern void dummy(void);            /* dummy function */

/*-----------------------------------------------------------------------*/
/*                                  vector table                         */
/*-----------------------------------------------------------------------*/
#pragma section VECT
const void (*const vect_tbl[])(void) =
{                          /* NO.  Offset       Exception Sources          */
     main,                 /* (000) H'00000000  Power-on reset PC          */
     (void(*)(void))0xFFFFFFFc, /* (001) H'00000004  Power-on Reset SP     */
     main,                      /* (002) H'00000008  Manual reset PC       */
     (void(*)(void))0xFFFFFFFc, /* (003) H'0000000C  Manual reset SP       */
     dummy,                /* (004) H'00000010  General illegal instruction*/
     dummy,                /* (005) H'00000014  (Reserved for system use)  */
```

Figure 2.2 Vector Definition File

```
    dummy,              /* (006) H'00000018  illegal slot instruction   */
    dummy,              /* (007) H'0000001C  (Reserved for system use)  */
    dummy,              /* (008) H'00000020  (Reserved for system use)  */
    dummy,              /* (009) H'00000024  CPU address error          */
    dummy,              /* (010) H'00000028  DMA address error          */
    dummy,              /* (011) H'0000002C  NMI                        */
    dummy,              /* (012) H'00000030  UBC (User break)           */
    dummy,              /* (013) H'00000034  (Reserved for system use)  */
    dummy,              /* (014) H'00000038  H-UDI                      */
    dummy,              /* (015) H'0000003C  (Reserved for system use)  */
    dummy,              /* (016) H'00000040  (Reserved for system use)  */
    dummy,              /* (017) H'00000044  (Reserved for system use)  */
    dummy,              /* (018) H'00000048  (Reserved for system use)  */
    dummy,              /* (019) H'0000004C  (Reserved for system use)  */
    dummy,              /* (020) H'00000050  (Reserved for system use)  */
    dummy,              /* (021) H'00000054  (Reserved for system use)  */
    dummy,              /* (022) H'00000058  (Reserved for system use)  */
    dummy,              /* (023) H'0000005C  (Reserved for system use)  */
    dummy,              /* (024) H'00000060  (Reserved for system use)  */
    dummy,              /* (025) H'00000064  (Reserved for system use)  */
    dummy,              /* (026) H'00000068  (Reserved for system use)  */
    dummy,              /* (027) H'0000006C  (Reserved for system use)  */
    dummy,              /* (028) H'00000070  (Reserved for system use)  */
    dummy,              /* (029) H'00000074  (Reserved for system use)  */
    dummy,              /* (030) H'00000078  (Reserved for system use)  */
    dummy,              /* (031) H'0000007C  (Reserved for system use)  */
    dummy,              /* (032) H'00000080  Trap inst (user vectors)   */
    dummy,              /* (033) H'00000084  Trap inst (user vectors)   */
    dummy,              /* (034) H'00000088  Trap inst (user vectors)   */
    dummy,              /* (035) H'0000008C  Trap inst (user vectors)   */
    dummy,              /* (036) H'00000090  Trap inst (user vectors)   */
    dummy,              /* (037) H'00000094  Trap inst (user vectors)   */
    dummy,              /* (038) H'00000098  Trap inst (user vectors)   */
    dummy,              /* (039) H'0000009C  Trap inst (user vectors)   */
    dummy,              /* (040) H'000000A0  Trap inst (user vectors)   */
    dummy,              /* (041) H'000000A4  Trap inst (user vectors)   */
    dummy,              /* (042) H'000000A8  Trap inst (user vectors)   */
    dummy,              /* (043) H'000000AC  Trap inst (user vectors)   */
    dummy,              /* (044) H'000000B0  Trap inst (user vectors)   */
    dummy,              /* (045) H'000000B4  Trap inst (user vectors)   */
    dummy,              /* (046) H'000000B8  Trap inst (user vectors)   */
    dummy,              /* (047) H'000000BC  Trap inst (user vectors)   */
    dummy,              /* (048) H'000000C0  Trap inst (user vectors)   */
    dummy,              /* (049) H'000000C4  Trap inst (user vectors)   */
    dummy,              /* (050) H'000000C8  Trap inst (user vectors)   */
    dummy,              /* (051) H'000000CC  Trap inst (user vectors)   */
    dummy,              /* (052) H'000000D0  Trap inst (user vectors)   */
```

Figure 2.2 Vector Definition File (cont)

```
    dummy,              /* (053) H'000000D4  Trap inst (user vectors)   */
    dummy,              /* (054) H'000000D8  Trap inst (user vectors)   */
    dummy,              /* (055) H'000000DC  Trap inst (user vectors)   */
    dummy,              /* (056) H'000000E0  Trap inst (user vectors)   */
    dummy,              /* (057) H'000000E4  Trap inst (user vectors)   */
    dummy,              /* (058) H'000000E8  Trap inst (user vectors)   */
    dummy,              /* (059) H'000000EC  Trap inst (user vectors)   */
    dummy,              /* (060) H'000000F0  Trap inst (user vectors)   */
    dummy,              /* (061) H'000000F4  Trap inst (user vectors)   */
    dummy,              /* (062) H'000000F8  Trap inst (user vectors)   */
    dummy,              /* (063) H'000000FC  Trap inst (user vectors)   */
    dummy,              /* (064) H'00000100  IRQ0                       */
    dummy,              /* (065) H'00000104  IRQ1                       */
    dummy,              /* (066) H'00000108  IRQ2                       */
    dummy,              /* (067) H'0000010C  IRQ3                       */
    dummy,              /* (068) H'00000110  IRQ4                       */
    dummy,              /* (069) H'00000114  IRQ5                       */
    dummy,              /* (070) H'00000118  IRQ6                       */
    dummy,              /* (071) H'0000011C  IRQ7                       */
    dummy,              /* (072) H'00000120  DMAC/ DEI0                 */
    dummy,              /* (073) H'00000124       */
    dummy,              /* (074) H'00000128       */
    dummy,              /* (075) H'0000012C       */
    dummy,              /* (076) H'00000130  DMAC/ DEI1                 */
    dummy,              /* (077) H'00000134       */
    dummy,              /* (078) H'00000138       */
    dummy,              /* (079) H'0000013C       */
    dummy,              /* (080) H'00000140  DMAC/ DEI2                 */
    dummy,              /* (081) H'00000144       */
    dummy,              /* (082) H'00000148       */
    dummy,              /* (083) H'0000014C       */
    dummy,              /* (084) H'00000150  DMAC/ DEI3                 */
    dummy,              /* (085) H'00000154       */
    dummy,              /* (086) H'00000158       */
    dummy,              /* (087) H'0000015C       */
    dummy,              /* (088) H'00000160  MTU0/TGIA_0                */
    dummy,              /* (089) H'00000164  MTU0/TGIB_0                */
    dummy,              /* (090) H'00000168  MTU0/TGIC_0                */
    dummy,              /* (091) H'0000016C  MTU0/TGID_0                */
    dummy,              /* (092) H'00000170  MTU0/TCIV_0                */
    dummy,              /* (093) H'00000174       */
    dummy,              /* (094) H'00000178       */
    dummy,              /* (095) H'0000017C       */
    dummy,              /* (096) H'00000180  MTU1/TGIA_1                */
    dummy,              /* (097) H'00000184  MTU1/TGIB_1                */
    dummy,              /* (098) H'00000188       */
    dummy,              /* (099) H'0000018C       */
```

Figure 2.2 Vector Definition File (cont)

```
    dummy,              /* (100) H'00000190  MTU1/TCIV_1              */
    dummy,              /* (101) H'00000194  MTU1/TCIU_1              */
    dummy,              /* (102) H'00000198       */
    dummy,              /* (103) H'0000019C       */
    dummy,              /* (104) H'000001A0  MTU2/TGIA_2              */
    dummy,              /* (105) H'000001A4  MTU2/TGIB_2              */
    dummy,              /* (106) H'000001A8       */
    dummy,              /* (107) H'000001AC       */
    dummy,              /* (108) H'000001B0  MTU2/TCIV_2              */
    dummy,              /* (109) H'000001B4  MTU2/TCIU_2              */
    dummy,              /* (110) H'000001B8       */
    dummy,              /* (111) H'000001BC       */
    dummy,              /* (112) H'000001C0  MTU3/TGIA_3              */
    dummy,              /* (113) H'000001C4  MTU3/TGIB_3              */
    dummy,              /* (114) H'000001C8  MTU3/TGIC_3              */
    dummy,              /* (115) H'000001CC  MTU3/TGID_3              */
    dummy,              /* (116) H'000001D0  MTU3/TCIV_3              */
    dummy,              /* (117) H'000001D4       */
    dummy,              /* (118) H'000001D8       */
    dummy,              /* (119) H'000001DC       */
    dummy,              /* (120) H'000001E0  MTU4/TGIA_4              */
    dummy,              /* (121) H'000001E4  MTU4/TGIB_4              */
    dummy,              /* (122) H'000001E8  MTU4/TGIC_4              */
    dummy,              /* (123) H'000001EC  MTU4/TGID_4              */
    dummy,              /* (124) H'000001F0  MTU4/TCIV_4              */
    dummy,              /* (125) H'000001F4       */
    dummy,              /* (126) H'000001F8       */
    dummy,              /* (127) H'000001FC       */
    dummy,              /* (128) H'00000200  SCI0/ERI                 */
    dummy,              /* (129) H'00000204  SCI0/RXI                 */
    dummy,              /* (130) H'00000208  SCI0/TXI                 */
    dummy,              /* (131) H'0000020C  SCI0/TEI                 */
    dummy,              /* (132) H'00000210  SCI1/ERI                 */
    dummy,              /* (133) H'00000214  SCI1/RXI                 */
    dummy,              /* (134) H'00000218  SCI1/TXI                 */
    dummy,              /* (135) H'0000021C  SCI1/TEI                 */
    dummy,              /* (136) H'00000220  A/D ADI0                 */
    dummy,              /* (137) H'00000224  A/D ADI1                 */
    dummy,              /* (138) H'00000228       */
    dummy,              /* (139) H'0000022C       */
    dummy,              /* (140) H'00000230  DTC/SWDTEND              */
    dummy,              /* (141) H'00000234       */
    dummy,              /* (142) H'00000238       */
    dummy,              /* (143) H'0000023C       */
    dummy,              /* (144) H'00000240  CMT/CMT0                 */
    dummy,              /* (145) H'00000244       */
    dummy,              /* (146) H'00000248       */
```

Figure 2.2 Vector Definition File (cont)

```
    dummy,              /* (147) H'0000024C       */
    dummy,              /* (148) H'00000250  CMT/CMT1                  */
    dummy,              /* (149) H'00000254       */
    dummy,              /* (150) H'00000258       */
    dummy,              /* (151) H'0000025C       */
    dummy,              /* (152) H'00000260  WDT/ITI                   */
    dummy,              /* (153) H'00000264  (Reserved for system use) */
    dummy,              /* (154) H'00000268       */
    dummy,              /* (155) H'0000026C       */
    dummy,              /* (156) H'00000270  I/O(MTU)/MTUOEI           */
    dummy,              /* (157) H'00000274       */
    dummy,              /* (158) H'00000278       */
    dummy,              /* (159) H'0000027C       */
    dummy,              /* (160) H'00000280       */
    dummy,              /* (161) H'00000284       */
    dummy,              /* (162) H'00000288       */
    dummy,              /* (163) H'0000028C       */
    dummy,              /* (164) H'00000290  (Reserved for system use) */
    dummy,              /* (165) H'00000294  (Reserved for system use) */
    dummy,              /* (166) H'00000298  (Reserved for system use) */
    dummy,              /* (167) H'0000029C  (Reserved for system use) */
    dummy,              /* (168) H'000002A0  SCI2/ERI                  */
    dummy,              /* (169) H'000002A4  SCI2/RXI                  */
    dummy,              /* (170) H'000002A8  SCI2/TXI                  */
    dummy,              /* (170) H'000002AC  SCI2/TEI                  */
    dummy,              /* (170) H'000002B0  SCI3/ERI                  */
    dummy,              /* (170) H'000002B4  SCI3/RXI                  */
    dummy,              /* (170) H'000002B8  SCI3/TXI                  */
    dummy,              /* (170) H'000002BC  SCI3/TEI                  */
    dummy,              /* (170) H'000002C0  (Reserved for system use) */
    dummy,              /* (170) H'000002C4  (Reserved for system use) */
    dummy,              /* (170) H'000002C8  (Reserved for system use) */
    dummy,              /* (170) H'000002CC  (Reserved for system use) */
    dummy,              /* (180) H'000002D0  (Reserved for system use) */
    dummy,              /* (180) H'000002D4  (Reserved for system use) */
    dummy,              /* (180) H'000002D8  (Reserved for system use) */
    dummy,              /* (180) H'000002DC  (Reserved for system use) */
    dummy,              /* (180) H'000002E0  (Reserved for system use) */
    dummy,              /* (180) H'000002E4  (Reserved for system use) */
    dummy,              /* (180) H'000002E8  (Reserved for system use) */
    dummy,              /* (180) H'000002EC  (Reserved for system use) */
    dummy,              /* (180) H'000002F0  (Reserved for system use) */
    dummy,              /* (180) H'000002F4  (Reserved for system use) */
    dummy,              /* (190) H'000002F8  (Reserved for system use) */
    dummy,              /* (191) H'000002FC  (Reserved for system use) */
    dummy,              /* (192) H'00000300  IIC/ICI                   */
    dummy,              /* (193) H'00000304  (Reserved for system use) */
```

Figure 2.2 Vector Definition File (cont)

```
    dummy,              /* (194) H'00000308  (Reserved for system use)  */
    dummy,              /* (195) H'0000030C  (Reserved for system use)  */
    dummy,              /* (196) H'00000310  (Reserved for system use)  */
    dummy,              /* (197) H'00000314  (Reserved for system use)  */
    dummy,              /* (198) H'00000318  (Reserved for system use)  */
    dummy,              /* (199) H'0000031C  (Reserved for system use)  */
    dummy,              /* (200) H'00000320  (Reserved for system use)  */
    dummy,              /* (201) H'00000324  (Reserved for system use)  */
    dummy,              /* (202) H'00000328  (Reserved for system use)  */
    dummy,              /* (203) H'0000032C  (Reserved for system use)  */
    dummy,              /* (204) H'00000330  (Reserved for system use)  */
    dummy,              /* (205) H'00000334  (Reserved for system use)  */
    dummy,              /* (206) H'00000338  (Reserved for system use)  */
    dummy,              /* (207) H'0000033C  (Reserved for system use)  */
    dummy,              /* (208) H'00000340  (Reserved for system use)  */
    dummy,              /* (209) H'00000344  (Reserved for system use)  */
    dummy,              /* (210) H'00000348  (Reserved for system use)  */
    dummy,              /* (211) H'0000034C  (Reserved for system use)  */
    dummy,              /* (212) H'00000350  (Reserved for system use)  */
    dummy,              /* (213) H'00000354  (Reserved for system use)  */
    dummy,              /* (214) H'00000358  (Reserved for system use)  */
    dummy,              /* (215) H'0000035C  (Reserved for system use)  */
    dummy,              /* (216) H'00000360  (Reserved for system use)  */
    dummy,              /* (217) H'00000364  (Reserved for system use)  */
    dummy,              /* (218) H'00000368  (Reserved for system use)  */
    dummy,              /* (219) H'0000036C  (Reserved for system use)  */
    dummy,              /* (220) H'00000370  (Reserved for system use)  */
    dummy,              /* (221) H'00000374  (Reserved for system use)  */
    dummy,              /* (222) H'00000378  (Reserved for system use)  */
    dummy,              /* (223) H'0000037C  (Reserved for system use)  */
    dummy,              /* (224) H'00000380  (Reserved for system use)  */
    dummy,              /* (225) H'00000384  (Reserved for system use)  */
    dummy,              /* (226) H'00000388  (Reserved for system use)  */
    dummy,              /* (227) H'0000038C  (Reserved for system use)  */
    dummy,              /* (228) H'00000390  (Reserved for system use)  */
    dummy,              /* (229) H'00000394  (Reserved for system use)  */
    dummy,              /* (230) H'00000398  (Reserved for system use)  */
    dummy,              /* (231) H'0000039C  (Reserved for system use)  */
    dummy,              /* (232) H'000003A0  (Reserved for system use)  */
    dummy,              /* (233) H'000003A4  (Reserved for system use)  */
    dummy,              /* (234) H'000003A8  (Reserved for system use)  */
    dummy,              /* (235) H'000003AC  (Reserved for system use)  */
    dummy,              /* (236) H'000003B0  (Reserved for system use)  */
    dummy,              /* (237) H'000003B4  (Reserved for system use)  */
    dummy,              /* (238) H'000003B8  (Reserved for system use)  */
    dummy,              /* (239) H'000003BC  (Reserved for system use)  */
    dummy,              /* (240) H'000003C0  (Reserved for system use)  */
```

Figure 2.2 Vector Definition File (cont)

```
    dummy,                /* (241) H'000003C4  (Reserved for system use)  */
    dummy,                /* (242) H'000003C8  (Reserved for system use)  */
    dummy,                /* (243) H'000003CC  (Reserved for system use)  */
    dummy,                /* (244) H'000003D0  (Reserved for system use)  */
    dummy,                /* (245) H'000003D4  (Reserved for system use)  */
    dummy,                /* (246) H'000003D8  (Reserved for system use)  */
    dummy,                /* (247) H'000003DC  (Reserved for system use)  */
};
```

Figure 2.2 Vector Definition File (cont)

2.1.3 Register Definition File

The SH7145F register definition file is shown in Appendix 3.1, SH7145F Register Definition File.

2.2 Single-Master Transmission

2.2.1 Specifications

(1) Ten bytes of data are written to EEPROM (HN58X2464, 64k bits, 8 words × 8 bits) using channel 0 of the SH7145F's I^2C Bus interface.

(2) The slave address of the connected EEPROM is [1010000], and data is written to EEPROM memory addresses H'0000 through H'0009.

(3) The write data is [H'01, H'02, H'03, H'04, H'05, H'06, H'07, H'08, H'09, H'0A].

(4) The devices connected to the I^2C Bus of this system are a master device (SH7145F) and a slave device (EEPROM) in a single-master configuration.

(5) The I^2C Bus data transfer clock frequency is 156 kHz.

(6) The SH7145F operating frequencies are 40 MHz for the CPU clock and 40 MHz for the on-chip peripheral clock.

(7) Figure 2.3 shows an example of connection between an SH7145F and EEPROM.

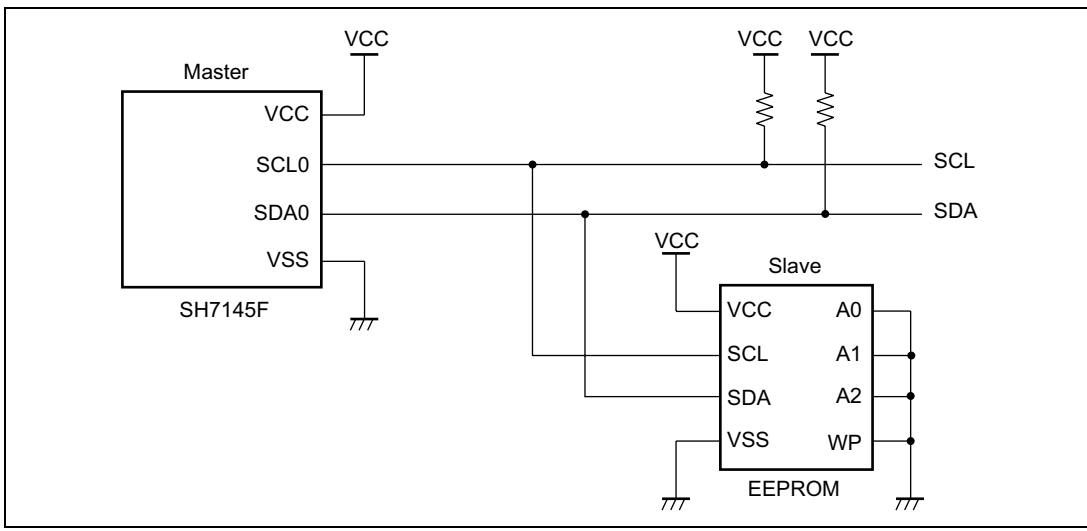
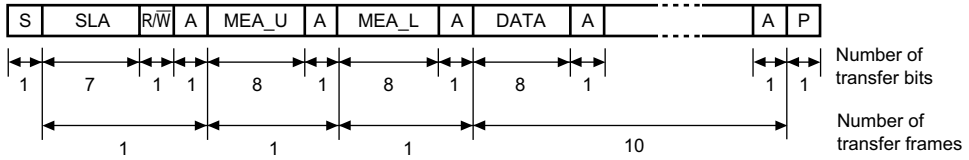


Figure 2.3 Example of Connection between SH7145F and EEPROM

The I^2C Bus format used in this sample task is shown in figure 2.4.

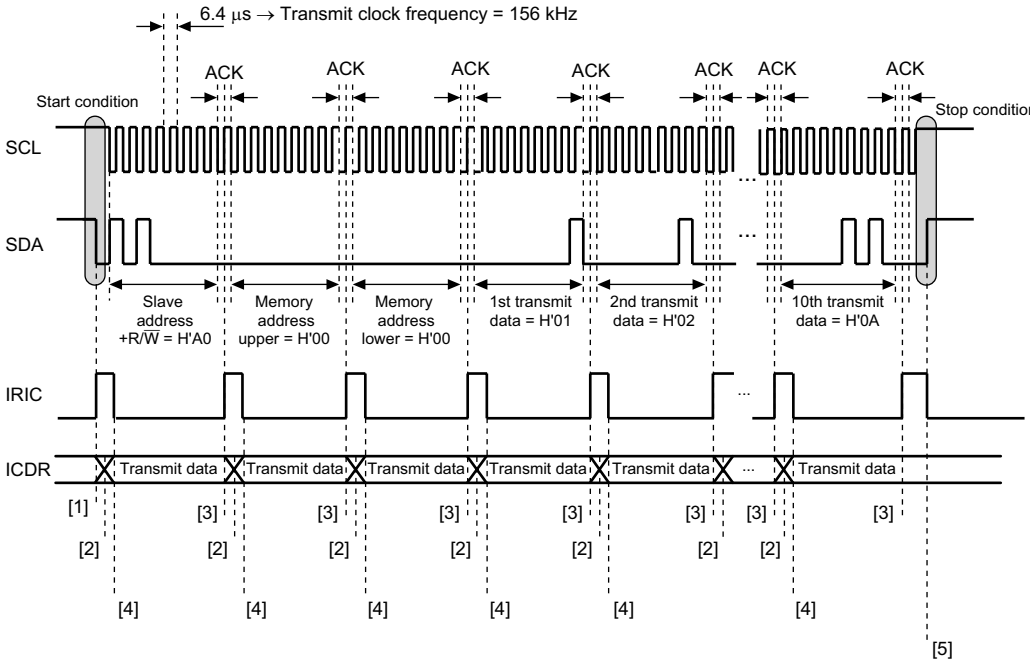


Legend:
S: Start condition
SLA: EEPROM slave address
R/$\overline{W}$: Transmission/reception direction
A: Acknowledge
MEA_U: EEPROM memory address (upper)
MEA_L: EEPROM memory address (lower)
DATA: Transmit data
P: Stop condition

Figure 2.4 Transfer Format Used in this Task

2.2.2 Operation

Figure 2.5 shows the principles of operation of this task.



	Software	Hardware
[1]	Start condition issuance (BBSY = 1, SCP = 0)	IRIC = 1, IRIR = 1 (flags set)
[2]	Write transmit data to ICDR0	No processing
[3]	No processing	IRIC = 1 at end of data transmission (at rise of 9th clock of transmit clock)
[4]	Clear IRIC to 0 for transmission end determination	No processing
[5]	Stop condition issuance (BBSY = 0, SCP = 0)	No processing

Figure 2.5 Principles of Operation of Single-Master Transmission

2.2.3 Software

(1) Modules

The modules used in this sample task are shown in the table below.

Table 2.1 Modules

Module Name	Label	Function
Main routine	main	I^2C initialization, pin setting
Dummy interrupt routine	dummy	Dummy interrupt handling
EEPROM write routine	Write_page_EEPROM	n-byte EEPROM write routine
Address setting routine	Set_adrs_EEPROM	Start condition generation, slave address issuance, EEPROM address setting

(2) Internal Registers Used

The internal registers used in this sample task are shown in the table below.

Table 2.2 Internal Registers Used

Register Name		Function	Address	Set Value
	Bit(s)		Bit(s)	
MSTCR1		Module standby control register 1	H'FFFF861C	
	MSTP21	I^2C module standby control bit Module standby cleared when MSTP21 = 0	Bit 5	B'0
PBCR1		Port B control register 1 Used to set port B pin functions in combination with port B control register 2	H'FFFF8398	H'0C00
PBCR2		Port B control register 2 Used to set port B pin functions in combination with port B control register 1 Sets port B3 (PB3) pin function as I^2C SDA0 I/O pin Sets port B2 (PB2) pin function as I^2C SCL0 I/O pin	H'FFFF839A	H'0000
ICDR0		I^2C Bus data register 8-bit readable/writable register, used as transmission data register when transmitting, and as reception data register when receiving.	H'FFFF880E	—

Register Name		Function	Address	Set Value
	Bit(s)		Bit(s)	
SAR0		Slave address register	H'FFFF880F	H'00
	SVA6-0	Slave address Unique address different from that of other slaves connected to I^2C Bus is set in bits SVA6 to SVA0.	Bits 7 to 1	
	FS	Format select Selects transfer format, together with FSX bit in SARX. Transfer format is I^2C Bus format when FS = FSX = 0.	Bit 0	
SARX0		Second slave address register	H'FFFF880E	H'00
	SVAX6-0	Second slave address Unique address different from that of other slaves connected to I^2C Bus is set in bits SVAX6 to SVAX0.	Bits 7 to 1	
	FSX	Format select Selects transfer format, together with FS bit in SAR. Transfer format is I^2C Bus format when FS = FSX = 0.	Bit 0	
ICMR0		I^2C Bus mode register	H'FFFF880F	H'38
	MLS	MSB-first/LSB-first selection MSB-first when MLS = 0	Bit 7	
	WAIT	Wait insertion bit Data and acknowledgment transferred continuously when WAIT = 0	Bit 6	
	CKS2 CKS1 CKS0	Transfer clock selection 2-0 Used to set transfer clock frequency in combination with IICX0 bit in SCRX register. 156 kHz (Pϕ = 40 MHz) when IICX = B'1, CKS[2:0] = B'111	Bit 5 Bit 4 Bit 3	
	BC2 BC1 BC0	Bit counter Used to set number of data bits to be transferred next in I^2C Bus format as 9 bits (including ACK bit)/frame. BC[2:0] = B'000	Bit 2 Bit 1 Bit 0	

Register Name		Function	Address	Set Value
	Bit(s)		Bit(s)	
ICCR0		I^2C Bus control register	H'FFFF8808	H'89
	ICE	I^2C Bus interface enable (ICE) When ICE = B'1, I^2C module is enabled for transfer, and ICMR and ICDR registers are valid.	Bit 7	
	IEIC	I^2C Bus interrupt enable Interrupt requests disabled when IEIC = B'0	Bit 6	
	MST	Master/slave selection Slave mode when MST = B'0	Bit 5	
	TRS	Transmission/reception selection Transmit mode when TES = B'0	Bit 4	
	ACKE	Acknowledge bit determination selection When ACKE = B'1, continuous transfer is suspended when acknowledge bit is 1.	Bit 3	
	BBSY	Busy bit Bus released state when BBSY = B'0	Bit 2	
	IRIC	I^2C Bus interface interrupt request flag Interrupt generated when IRIC = B'1	Bit 1	
	SCP	Start condition/stop condition issuance disable bit When SCP = B'0, issues start condition, stop condition in combination with BBSY flag.	Bit 0	
ICSR0		I^2C Bus status register	H'FFFF8809	—
	ESTP	Error stop condition detection flag	Bit 7	
	STOP	Normal stop condition detection flag	Bit 6	
	IRTR	I^2C Bus interface continuous transmission/ reception interrupt request flag	Bit 5	
	AASX	Second slave address recognition flag	Bit 4	
	AL	Arbitration lost flag	Bit 3	
	AAS	Slave address recognition flag	Bit 2	
	ADZ	General call address recognition flag	Bit 1	
	ACKB	Acknowledge bit Stores acknowledge data.	Bit 0	

Register Name		Function	Address	Set Value
	Bit(s)		Bit(s)	
SCRX		Serial control register X	H'FFFF87F0	H'39
	Reserved	Reserved bits Always read as 0. Write value should always be 0.	Bits 7, 6	
	IICX0	I^2C transfer rate select 0 Selects master mode transfer rate in combination with CKD[2:0] in ICMR. IICX0 = B'1	Bit 5	
	IICE	I^2C master enable When IICE = B'1, I^2C Bus interface register access is enabled.	Bit 4	
	HNDS	Handshake reception bit When HNDS = B'1, continuous reception operation is disabled.	Bit 3	
	Reserved	Reserved bit Always read as 0. Write value should always be 0.	Bit 2	
	ICDRF0	Indicates whether there is valid receive data in ICDR.	Bit 1	
	STOPIM	Stop condition detection interrupt mask When STOPIM = B'1, IRIC interrupt generation is suppressed in slave mode even if a stop condition is detected.	Bit 0	

(3) Variables

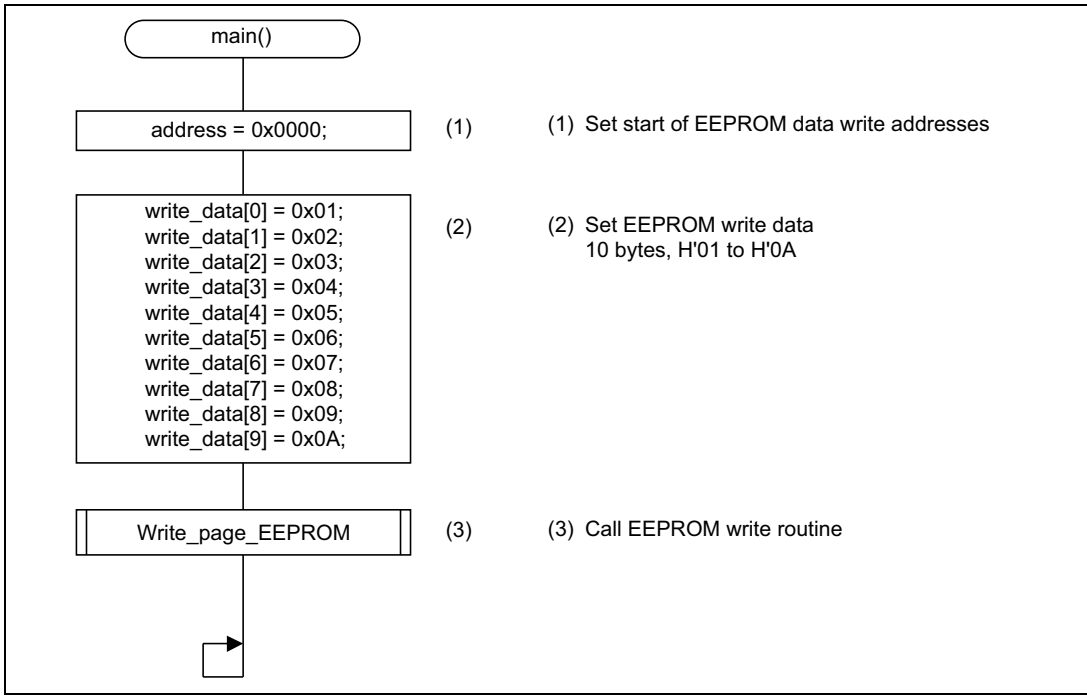
Variable	Function	Data Length	Initial Value	Module
write_data[0]	1st byte of transmit data	1 byte	H'01	Main routine
write_data[1]	2nd byte of transmit data	1 byte	H'02	Main routine
write_data[2]	3rd byte of transmit data	1 byte	H'03	Main routine
write_data[3]	4th byte of transmit data	1 byte	H'04	Main routine
write_data[4]	5th byte of transmit data	1 byte	H'05	Main routine
write_data[5]	6th byte of transmit data	1 byte	H'06	Main routine
write_data[6]	7th byte of transmit data	1 byte	H'07	Main routine
write_data[7]	8th byte of transmit data	1 byte	H'08	Main routine
write_data[8]	9th byte of transmit data	1 byte	H'09	Main routine
write_data[9]	10th byte of transmit data	1 byte	H'0A	Main routine
address	EEPROM write start address	2 bytes	H'0000	Main routine
adrs	EEPROM write start address copy	2 bytes	—	EEPROM write routine
num	Number of transmit data	1 byte	H'0A	EEPROM write routine
w_data	Pointer variable to transmit data array variable write_data	4 bytes	—	EEPROM write routine
ack	Acknowledge reception determination flag	1 byte	H'01	EEPROM write routine

(4) RAM Used

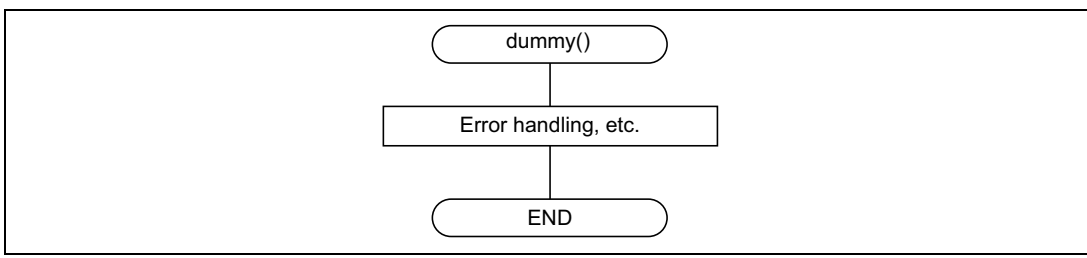
This sample task does not use any RAM apart from the variables.

2.2.4 Flowcharts

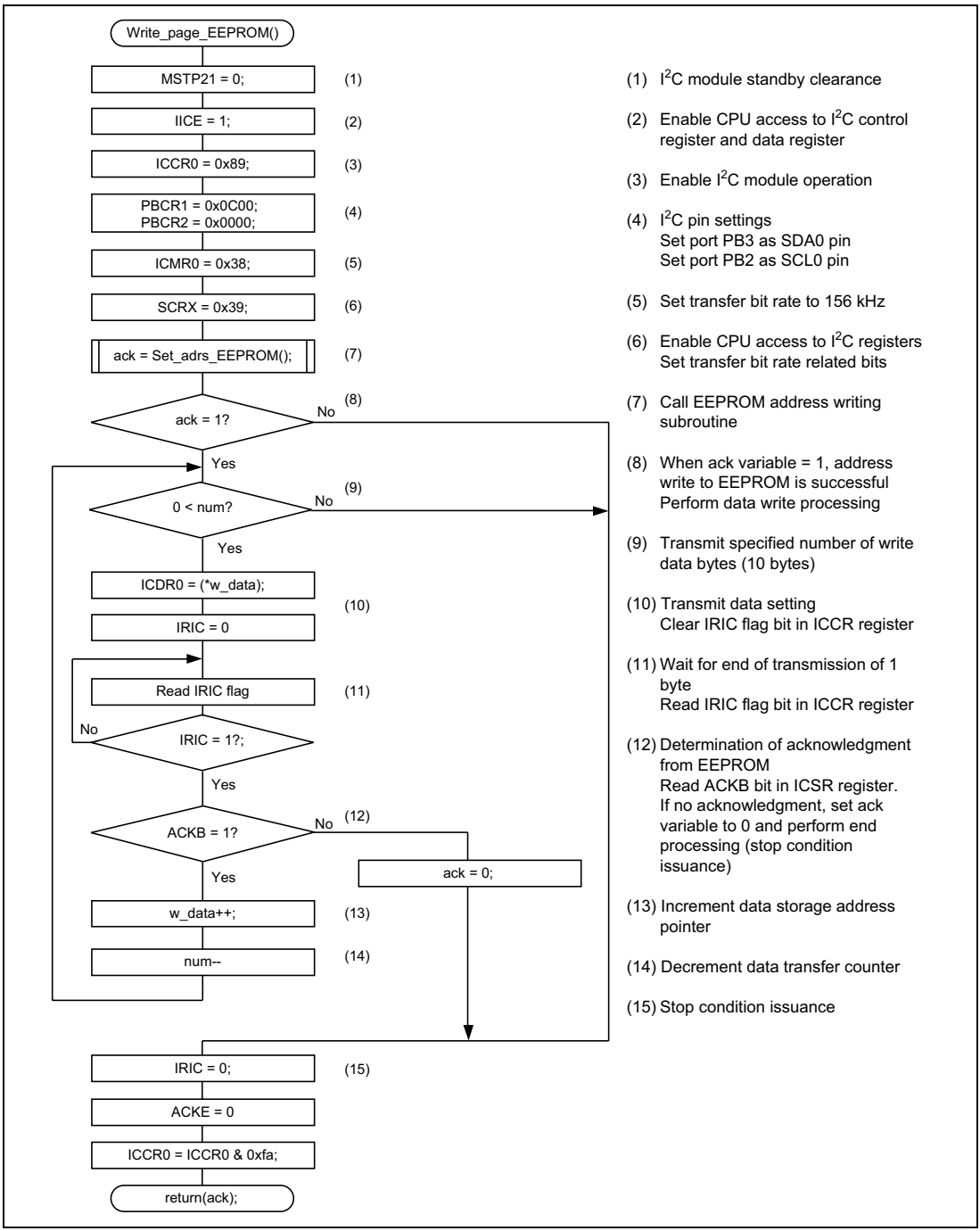
(1) Main routine



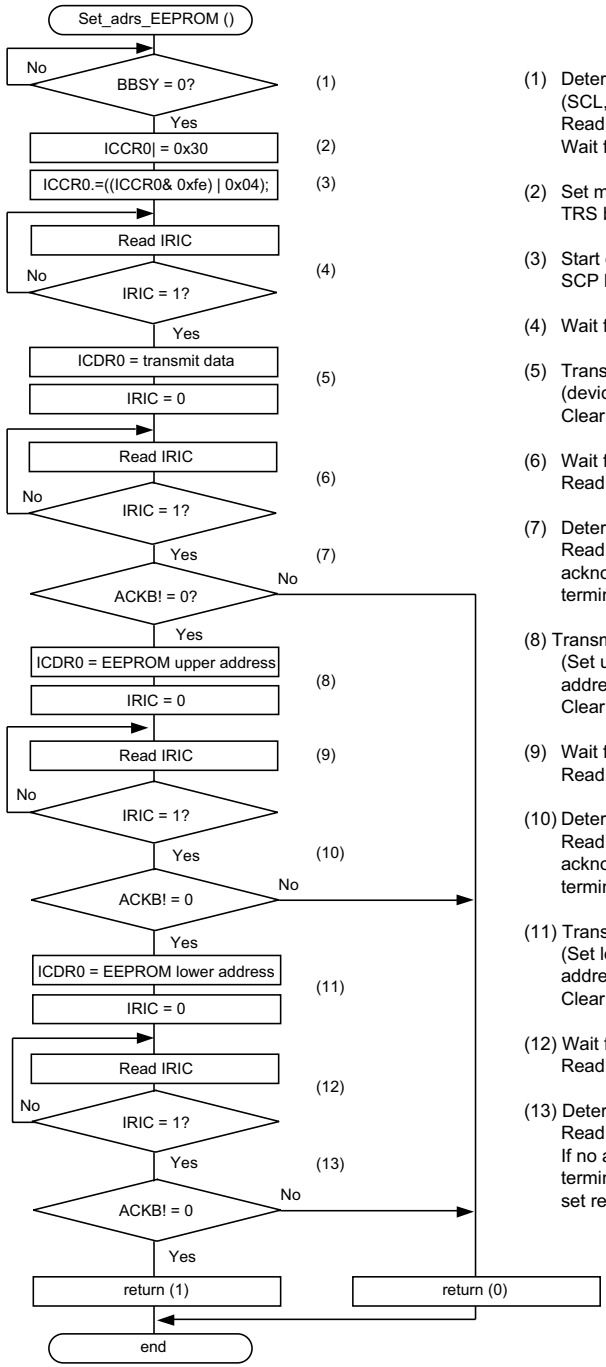
(2) Dummy interrupt routine



(3) EEPROM write subroutine



(4) Start condition issuance, slave address and EEPROM memory address transmission subroutine



(1) Determination of state of I^2C Bus lines (SCL, SDA)
Read BBSY flag bit in ICCR register
Wait for bus free state (BBSY = 0)

(2) Set master transmit mode (MST bit = 1, TRS bit = 1)

(3) Start condition issuance (BBSY bit = 1, SCP bit = 0)

(4) Wait for start condition generation

(5) Transmit data setting
(device code, slave address, write bits)
Clear IRIC flag bit in ICCR register

(6) Wait for end of transmission of 1 byte
Read IRIC flag bit in ICCR register

(7) Determination of acknowledgment from EEPROM
Read ACKB bit in ICSR register, and if no acknowledgment, set return value to 0 and terminate function

(8) Transmit data setting
(Set upper byte address of EEPROM write start address)
Clear IRIC flag bit in ICCR register

(9) Wait for end of transmission of 1 byte
Read IRIC flag bit in ICCR register

(10) Determination of acknowledgment from EEPROM
Read ACKB bit in ICSR register, and if no acknowledgment, set return value to 0 and terminate function

(11) Transmit data setting
(Set lower byte address of EEPROM write start address)
Clear IRIC flag bit in ICCR register

(12) Wait for end of transmission of 1 byte
Read IRIC flag bit in ICCR register

(13) Determination of acknowledgment from EEPROM
Read ACKB bit in ICSR register
If no acknowledgment, set return value to 0 and terminate function; if acknowledgment present, set return value to 1 and terminate function

2.2.5 Program Listing

```
//*********************************************************************
//   SH7144F Group -SH7145- I2C-bus Application Note
//         Single master transmit
//         n Byte data write/read 64kbit EEPROM
//                      Clock :CPU=40MHz  (External input=10MHz)
//                            :Peripheral=40MHz
//         I2c bit rate :156kHz
//         Written      :2003/2/1  Rev.2.0
//*********************************************************************
#include <machine.h>
#include "iodefine.h"    // SH7145  I/O Register Definition


//------------ Symbol Definition ------------------------------------------
#define DEVICE_CODE    0xa0             // EEPROM DEVICE CODE:b'1010
#define SLAVE_ADRS     0x00             // SLAVE ADRS:b'00
#define IIC_DATA_W     0x00             // WRITE DATA:b'0
#define IIC_DATA_R     0x01             // READ DATA:b'1
#define DATA_NUM       10               // data size


//------------ Function Definition ----------------------------------------
void main(void);
void dummy(void);

unsigned char Write_page_EEPROM(unsigned short,unsigned char*,unsigned char);
unsigned char Set_adrs_EEPROM(unsigned short);


//*********************************************************************
//    main
//*********************************************************************
void main(void)
{
      unsigned short    address;         // EEPROM memory address

      address= 0x0000;                   // set EEPROM address


      // set write data
      write_data[0]=0x01;
      write_data[1]=0x02;
      write_data[2]=0x03;
      write_data[3]=0x04;
      write_data[4]=0x05;
      write_data[5]=0x06;
      write_data[6]=0x07;
      write_data[7]=0x08;
      write_data[8]=0x09;
```

```
      write_data[9]=0x0a;

      // EEPROM data write
      Write_page_EEPROM(address,write_data,DATA_NUM);

      while(1);


}



//*********************************************************
//   dummy interrupt function
//*********************************************************
#pragma interrupt(dummy)
void dummy(void)
{
      //  Interrupt error
}




//*****************************************************************
//   Write_page_EEPROM
//         argument1  ;write address(unsigned short)
//         argument2  ;write data(unsigned char)
//         argument3  ;write data number(unsigned char)
//         return     ;1=OK/0=NG EEPROM NO_ACK(unsigned char)
//*****************************************************************
unsigned char Write_page_EEPROM(unsigned short adrs,unsigned char*
w_data,unsigned char num)
{
      unsigned char ack;               // ACK check flag

      // Set standby mode
      P_STBY.MSTCR1.BIT.MSTP21 = 0;    // disable I2C standby mode

      ack = 1;
      P_IIC.SCRX.BIT.IICE = 1;         // Enables CPU access to the register

      P_IIC.ICCR0.BYTE = 0x89;
                  // ICE(7)=b'1        Enable I2C bus interface
                  // IEIC(6)=b'0       Disables the interrupt
                  // MST(5)=b'0        Slave mode
                  // TRS(4)=b'0        Receive mode
                  // ACKE(3)=b'1       Continuous data transfer is halted
                  // BBSY(2)=b'0
                  // IRIC(1)=b'0
                  // SCP(0)=b'1        Start/stop condition issuance disabling
```

```
      // set I2C pin function
      P_PORTB.PBCR1.WORD = 0x0c00;
                           // SDA0(PB3-32pin@SH7145F),SCL0(PB2-31pin@SH7145F)
      P_PORTB.PBCR2.WORD = 0x0000;

      P_IIC.ICMR0.BYTE = 0x38;
                  // MILS(7)=b'0  MSB first
                  // WAIT(6)=b'0  A wait state is inserted between DATA and ACK
                  // CKS2[2:0](5:3)=b'111   Transfer clock select
                  //               156kHz@(@P-fai=40MHz,IICX=1)
                  //              39.1kHz@(@P-fai=10MHz,IICX=1)
      P_IIC.SCRX.BYTE = 0x39;
                     // IICX(5)=b'1 transfer-rate select, reference CKS bit
                     // IICE(4)=b'1 Enables CPU access to the register
                     // HNDS(3)=b'1
                     // STOPIM(0)=b'1    disables interrupt requests


      // Set device code,EEPROM address
      ack = Set_adrs_EEPROM(adrs);

      // EEPROM write data Transmission (n byte)
      if( ack==1 ){
           for( ; 0<num; num-- ){
              P_IIC.ICDR0.BYTE = (*w_data);      // write data set
              P_IIC.ICCR0.BIT.IRIC = 0;          // clear IRIC
              while( P_IIC.ICCR0.BIT.IRIC==0 );  // Wait 1byte transmitted
              if( P_IIC.ICSR0.BIT.ACKB != 0 ){   // Test the acknowledge bit
                   ack = 0;                      // no ACK
                   break;
              }
              w_data++;                          // write data pointer increment
           }
      }

      // Stop condition issuance
      P_IIC.ICCR0.BIT.IRIC = 0;                    // clear IRIC
      P_IIC.ICCR0.BIT.ACKE = 0;                    // set AKCE=0
      P_IIC.ICCR0.BYTE = P_IIC.ICCR0.BYTE & 0xfa;  // Stop condition
issuance(BBSY=0,SCP=0)

      return(ack);
}
```

```
//*********************************************************************
//   Set_adrs_EEPROM
//         argument1   ;write address(unsigned short)
//         return      ;1=OK/0=NG EEPROM NO_ACK(unsigned char)
//*********************************************************************
unsigned char Set_adrs_EEPROM(unsigned short adrs)
{

      while( P_IIC.ICCR0.BIT.BBSY!=0 );  // BUS FREE?(BBSY==0→Bus Free)


      // Master-Transmission,Generate the start condition.
      P_IIC.ICCR0.BYTE |= 0x30;     // Select master transmit mode(MST=1,TRS=1)

      P_IIC.ICCR0.BYTE=((P_IIC.ICCR0.BYTE & 0xfe) | 0x04);
                                     // Generate start condition(BBSY=1,SCP=0)
      while( P_IIC.ICCR0.BIT.IRIC==0 );
                                     // Wait for a start condition generation


      // Slave address+W Transmission
      P_IIC.ICDR0.BYTE = (unsigned char)(DEVICE_CODE|SLAVE_ADRS|IIC_DATA_W);
                                                 // data set
      P_IIC.ICCR0.BIT.IRIC = 0;                  // clear IRIC
      while( P_IIC.ICCR0.BIT.IRIC==0 );          // Wait 1byte transmitted
      if( P_IIC.ICSR0.BIT.ACKB!=0 ){             // Test the acknowledge bit
      return (0);                                // no ACK
      }


      // EEPROM upper address Transmission(1byte)
      P_IIC.ICDR0.BYTE = (unsigned char)(adrs>>8); // data set
      P_IIC.ICCR0.BIT.IRIC = 0;                  // clear IRIC
      while( P_IIC.ICCR0.BIT.IRIC==0 );          // Wait 1byte transmitted
      if( P_IIC.ICSR0.BIT.ACKB!=0 ){             // Test the acknowledge bit
      return (0);                                // no ACK
      }


      // EEPROM lower address Transmission(1byte)
      P_IIC.ICDR0.BYTE = (unsigned char)(adrs & 0x00ff);  // data set
      P_IIC.ICCR0.BIT.IRIC = 0;                  // clear IRIC
      while( P_IIC.ICCR0.BIT.IRIC==0 );          // Wait 1byte transmitted
      if( P_IIC.ICSR0.BIT.ACKB!=0 ){             // Test the acknowledge bit
      return (0);                                // no ACK
      }

      return (1);                                // ACK OK

}
```

2.3 Single-Master Reception

2.3.1 Specifications

(1) Ten bytes of data are read from EEPROM (HN58X2464, 64k bits, 8 words × 8 bits) using channel 0 of the SH7145F's I^2C Bus interface.

(2) The slave address of the connected EEPROM is [1010000], and data in EEPROM memory addresses H'0000 through H'0009 is read.

(3) The read data is captured in a variable array.

(4) The devices connected to the I^2C Bus of this system are a master device (SH7145F) and a slave device (EEPROM) in a single-master configuration.

(5) The I^2C Bus data transfer clock frequency is 156 kHz.

(6) The SH7145F operating frequencies are 40 MHz for the CPU clock and 40 MHz for the on-chip peripheral clock.

(7) Figure 2.6 shows an example of connection between an SH7145F and EEPROM.

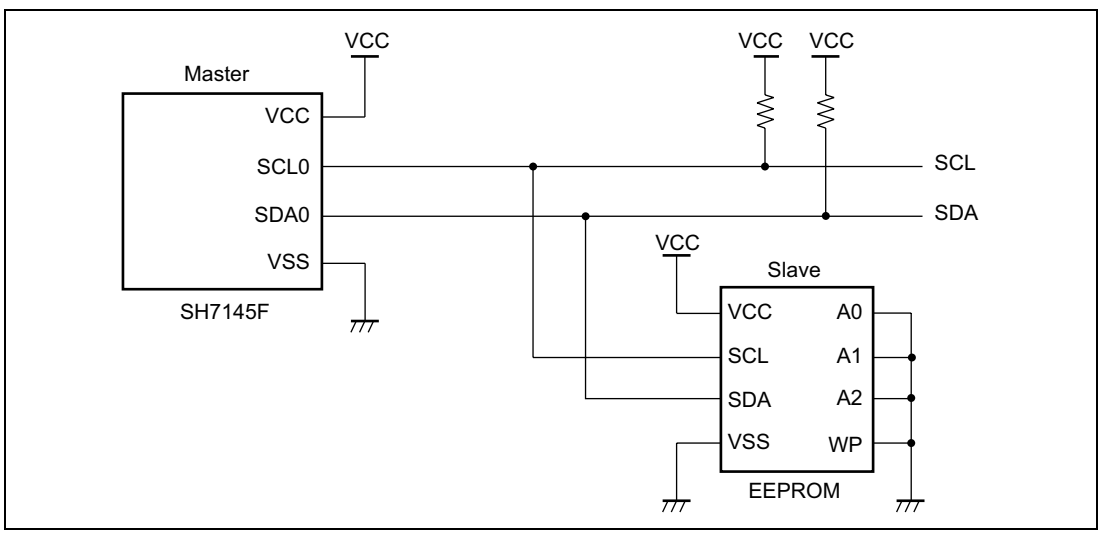
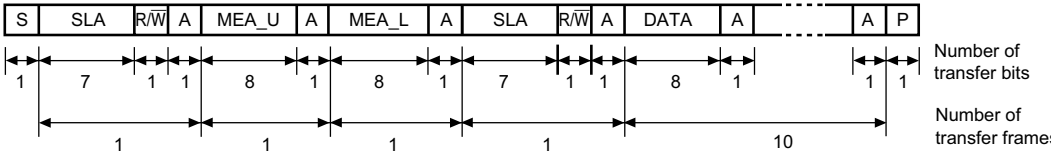


Figure 2.6 Example of Connection between SH7145F and EEPROM

The I^2C Bus format used in this sample task is shown in figure 2.7.

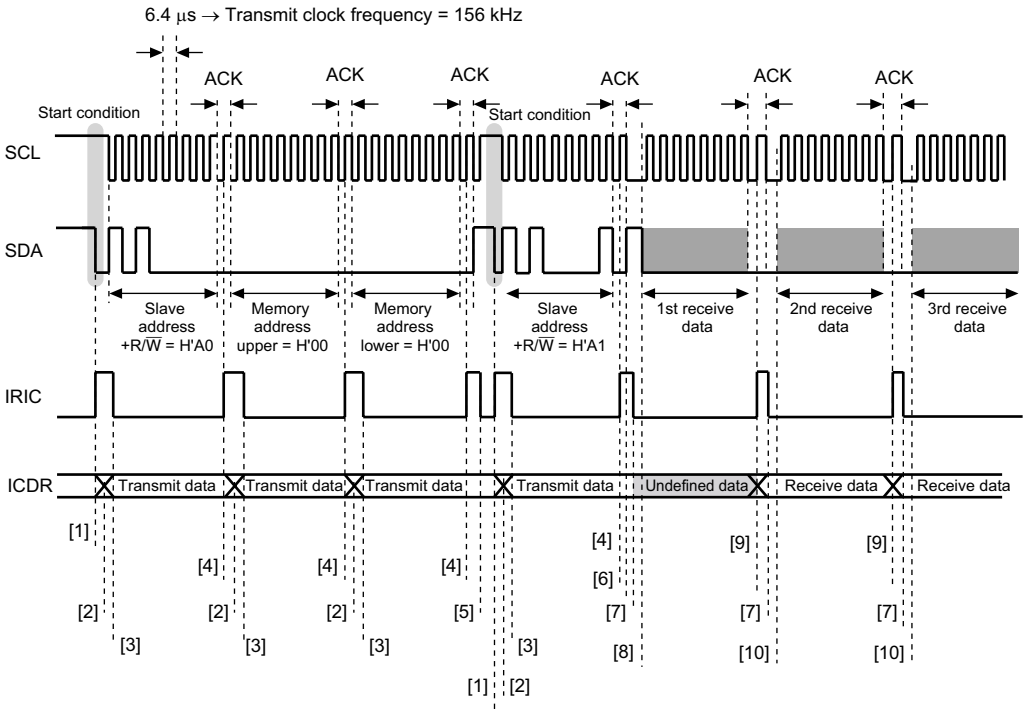


Legend:
S: Start condition
SLA: EEPROM slave address
R/$\overline{W}$: Transmission/reception direction
A: Acknowledge
MEA_U: EEPROM memory address (upper)
MEA_L: EEPROM memory address (lower)
DATA: Transmit data
P: Stop condition

Figure 2.7 Transfer Format Used in this Task

2.3.2 Operation

Figure 2.8 shows the principles of operation of this task.



	Software	Hardware
[1]	Start condition issuance (BBSY = 1, SCP = 0)	Flag setting, IRIC = 1, IRTR = 1
[2]	Write transmit data to ICDR0	No processing
[3]	Clear IRIC to 0 for transmission end determination	No processing
[4]	No processing	IRIC = 1 (end of data transmission, rise of 9th clock of transmit clock)
[5]	Clear IRIC to 0 for start condition detection determination	No processing
[6]	Set master receive mode (MST = 1, TRS = 0, HNDS = 1, WAIT = 0)	No processing
[7]	Clear ICIR flag to 0	No processing
[8]	Dummy read of ICDR0	Start of reception
[9]	No processing	Drive SDA low to return acknowledgment IRIC = 1 (rise of 9th clock of receive clock)
[10]	Read receive data from ICDR0	Start of reception

Figure 2.8 Principles of Operation of Single-Master Reception

2.3.3 Software

(1) Modules

The modules used in this sample task are shown in the table below.

Table 2.3 Modules

Module Name	Label	Function
Main routine	main	I^2C initialization, pin setting
Dummy interrupt routine	dummy	Dummy interrupt handling
EEPROM read routine	Read_page_EEPROM	n-byte EEPROM read routine (n > 1)
Address setting routine	Set_adrs_EEPROM	Start condition generation, slave address issuance, EEPROM address setting

(2) Internal Registers Used

The internal registers used in this sample task are shown in the table below.

Table 2.4 Internal Registers Used

Register Name		Function	Address	Set Value
	Bit(s)		Bit(s)	
MSTCR1		Module standby control register 1	H'FFFF861C	
	MSTP21	I^2C module standby control bit Module standby cleared when MSTP21 = 0	Bit 5	B'0
PBCR1		Port B control register 1 Sets port B pin functions in combination with port B control register 2	H'FFFF8398	H'0C00
PBCR2		Port B control register 2 Used to set port B pin functions in combination with port B control register 1 Sets port B3 (PB3) pin function as I^2C SDA0 I/O pin Sets port B2 (PB2) pin function as I^2C SCL0 I/O pin	H'FFFF839A	H'0000
ICDR0		I^2C Bus data register 8-bit readable/writable register, used as transmission data register when transmitting, and as reception data register when receiving.	H'FFFF880E	—

Register Name		Function	Address	Set Value
	Bit(s)		Bit(s)	
SAR0		Slave address register	H'FFFF880F	H'00
	SVA6-0	Slave address Unique address different from that of other slaves connected to I^2C Bus is set in bits SVA6 to SVA0.	Bits 7 to 1	
	FS	Format select Selects transfer format, together with FSX bit in SARX. Transfer format is I^2C Bus format when FS = FSX = 0.	Bit 0	
SARX0		Second slave address register	H'FFFF880E	H'00
	SVAX6-0	Second slave address Unique address different from that of other slaves connected to I^2C Bus is set in bits SVAX6 to SVAX0.	Bits 7 to 1	
	FSX	Format select Selects transfer format, together with FS bit in SAR. Transfer format is I^2C Bus format when FS = FSX = 0.	Bit 0	
ICMR0		I^2C Bus mode register	H'FFFF880F	H'38
	MLS	MSB-first/LSB-first selection MSB-first when MLS = 0	Bit 7	
	WAIT	Wait insertion bit Data and acknowledgment transferred continuously when WAIT = 0	Bit 6	
	CKS2 CKS1 CKS0	Transfer clock selection 2-0 Used to set transfer clock frequency in combination with IICX0 bit in SCRX register. 156 kHz (Pϕ = 40 MHz) when IICX = B'1, CKS[2:0] = B'111	Bit 5 Bit 4 Bit 3	
	BC2 BC1 BC0	Bit counter Used to set number of data bits to be transferred next in I^2C Bus format as 9 bits (including ACK bit)/frame. BC[2:0] = B'000	Bit 2 Bit 1 Bit 0	

Register Name		Function	Address	Set Value
	Bit(s)		Bit(s)	
ICCR0		I^2C Bus control register	H'FFFF8808	H'89
	ICE	I^2C Bus interface enable (ICE) When ICE = B'1, I^2C module is enabled for transfer, and ICMR and ICDR registers are valid.	Bit 7	
	IEIC	I^2C Bus interrupt enable Interrupt requests disabled when IEIC = B'0	Bit 6	
	MST	Master/slave selection Slave mode when MST = B'0	Bit 5	
	TRS	Transmission/reception selection Receive mode when TES = B'0	Bit 4	
	ACKE	Acknowledge bit determination selection When ACKE = B'1, continuous transfer is suspended when acknowledge bit is 1.	Bit 3	
	BBSY	Busy bit Bus released state when BBSY = B'0	Bit 2	
	IRIC	I^2C Bus interface interrupt request flag Interrupt generated when IRIC = B'1	Bit 1	
	SCP	Start condition/stop condition issuance disable bit When SCP = B'0, issues start condition, stop condition in combination with BBSY flag.	Bit 0	
ICSR0		I^2C Bus status register	H'FFFF8809	—
	ESTP	Error stop condition detection flag	Bit 7	
	STOP	Normal stop condition detection flag	Bit 6	
	IRTR	I^2C Bus interface continuous transmission/reception interrupt request flag	Bit 5	
	AASX	Second slave address recognition flag	Bit 4	
	AL	Arbitration lost flag	Bit 3	
	AAS	Slave address recognition flag	Bit 2	
	ADZ	General call address recognition flag	Bit 1	
	ACKB	Acknowledge bit Stores acknowledge data.	Bit 0	

Register Name		Function	Address	Set Value
	Bit(s)		Bit(s)	
SCRX		Serial control register X	H'FFFF87F0	H'39
	Reserved	Reserved bits Always read as 0. Write value should always be 0.	Bits 7, 6	
	IICX0	I^2C transfer rate select 0 Selects master mode transfer rate in combination with CKD[2:0] in ICMR. IICX0 = B'1	Bit 5	
	IICE	I^2C master enable When IICE = B'1, I^2C Bus interface register access is enabled.	Bit 4	
	HNDS	Handshake reception bit When HNDS = B'1, continuous reception operation is disabled.	Bit 3	
	Reserved	Reserved bit Always read as 0. Write value should always be 0.	Bit 2	
	ICDRF0	Indicates whether there is valid receive data in ICDR.	Bit 1	
	STOPIM	Stop condition detection interrupt mask When STOPIM = B'1, IRIC interrupt generation is suppressed in slave mode even if a stop condition is detected.	Bit 0	

(3) Variables

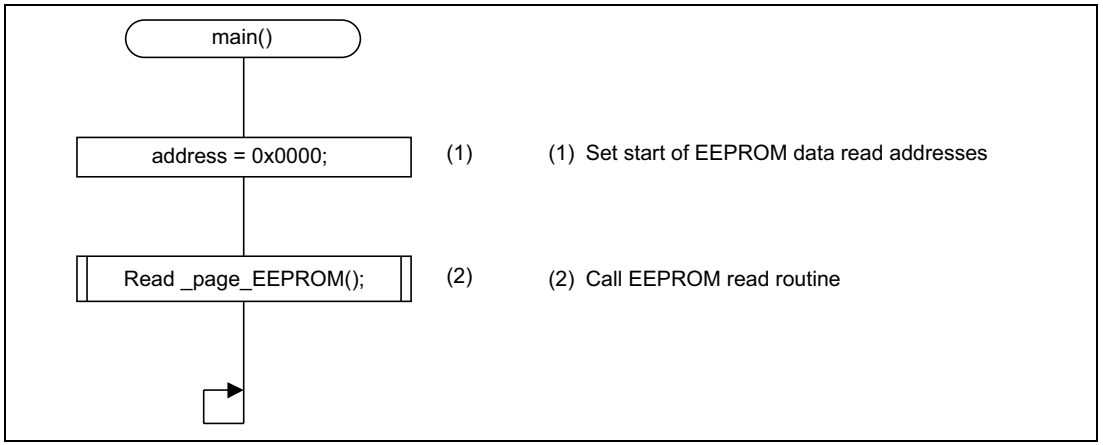
Variable	Function	Data Length	Initial Value	Module
read_data[0]	1st byte of receive data	1 byte	—	Main routine
read_data[1]	2nd byte of receive data	1 byte	—	Main routine
read_data[2]	3rd byte of receive data	1 byte	—	Main routine
read_data[3]	4th byte of receive data	1 byte	—	Main routine
read_data[4]	5th byte of receive data	1 byte	—	Main routine
read_data[5]	6th byte of receive data	1 byte	—	Main routine
read_data[6]	7th byte of receive data	1 byte	—	Main routine
read_data[7]	8th byte of receive data	1 byte	—	Main routine
read_data[8]	9th byte of receive data	1 byte	—	Main routine
read_data[9]	10th byte of receive data	1 byte	—	Main routine
address	EEPROM read start address	2 bytes	H'0000	Main routine
adrs	EEPROM read start address copy	2 bytes	—	EEPROM read routine
num	Number of receive data	1 byte	H'0A	EEPROM read routine
r_data	Pointer variable to receive data array variable read_data	4 bytes	—	EEPROM read routine
dummy	Dummy read data	1 byte	—	EEPROM read routine
ack	Acknowledge reception determination flag	1 byte	H'01	EEPROM read routine

(4) RAM Used

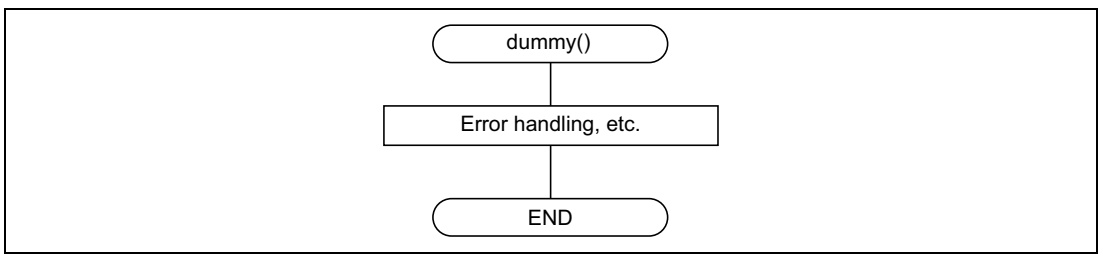
This sample task does not use any RAM apart from the variables.

2.3.4 Flowcharts

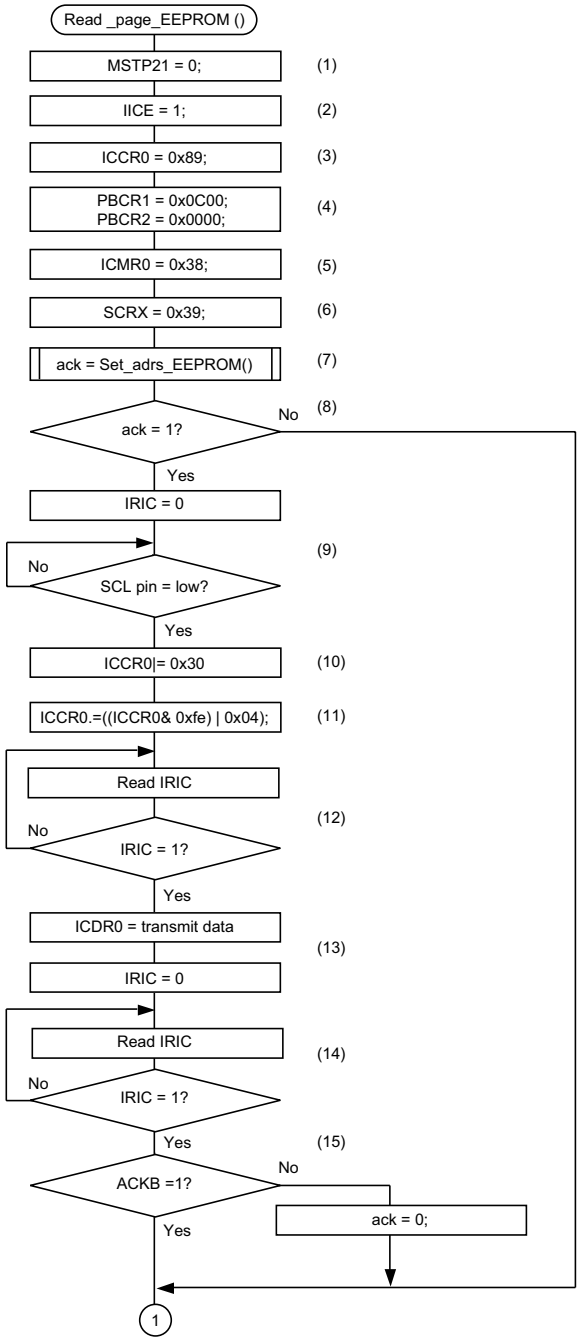
(1) Main routine



(2) Dummy interrupt routine



(3) EEPROM read subroutine



(1) I^2C module standby clearance
Set module standby bit to 0

(2) Enable CPU access to I^2C registers

(3) Enable I^2C module operation

(4) I^2C pin settings
Set port PB3 as SDA0 pin
Set port PB2 as SCL0 pin

(5) Set transfer bit rate to 156 kHz

(6) Enable CPU access to I^2C registers
Set transfer bit rate related bits

(7) Call EEPROM address writing subroutine

(8) When ack variable = 1, address write to EEPROM is successful
Perform data write processing

(9) Clear IRIC flag bit in ICCR register
Wait until SCL pin goes low in order to perform confirmation of start condition issuance for retransmission

(10) Set master transmit mode
MST bit = 1, TRS bit = 1

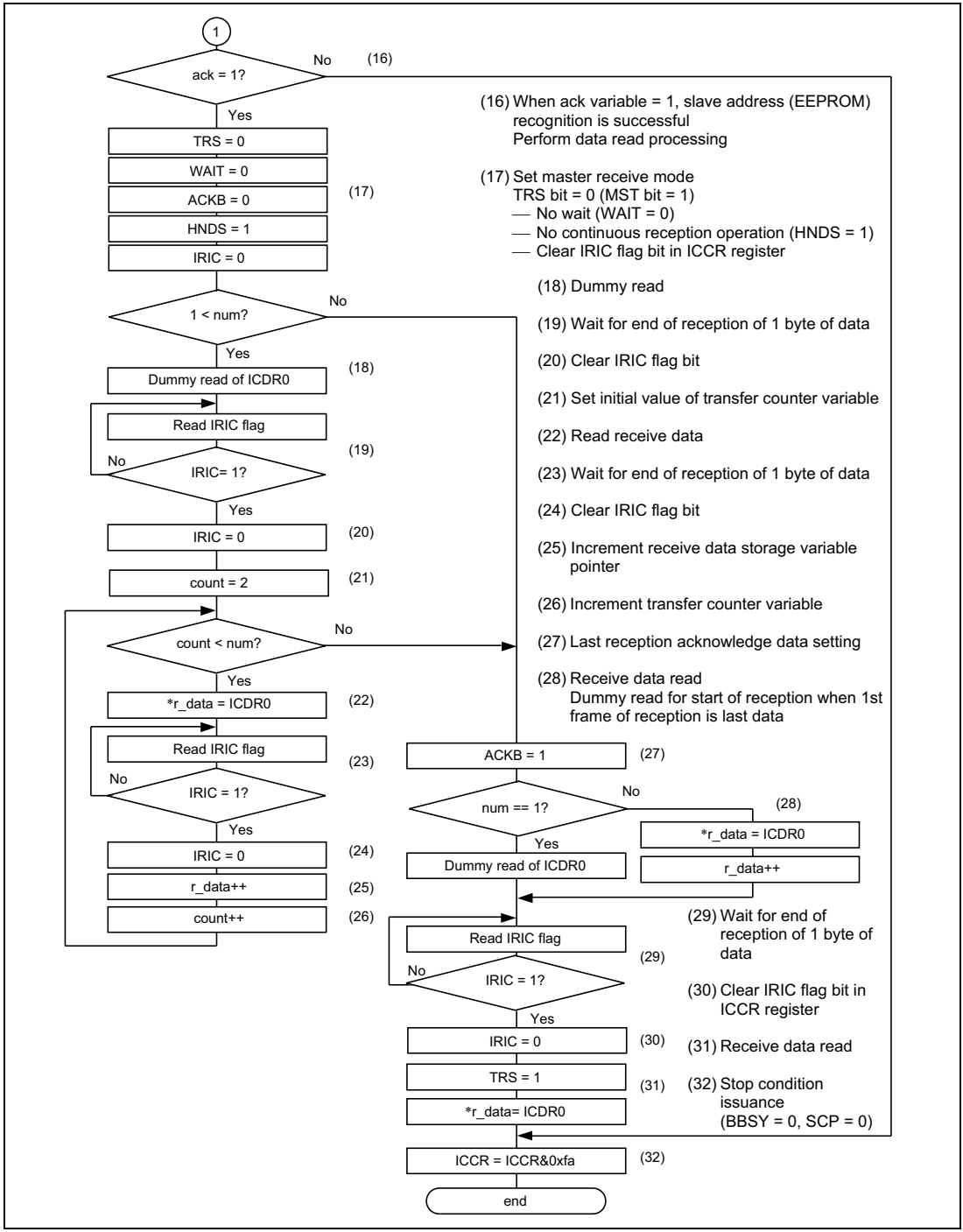
(11) Start condition issuance
BBSY bit = 1, SCP bit = 0

(12) Wait for start condition generation

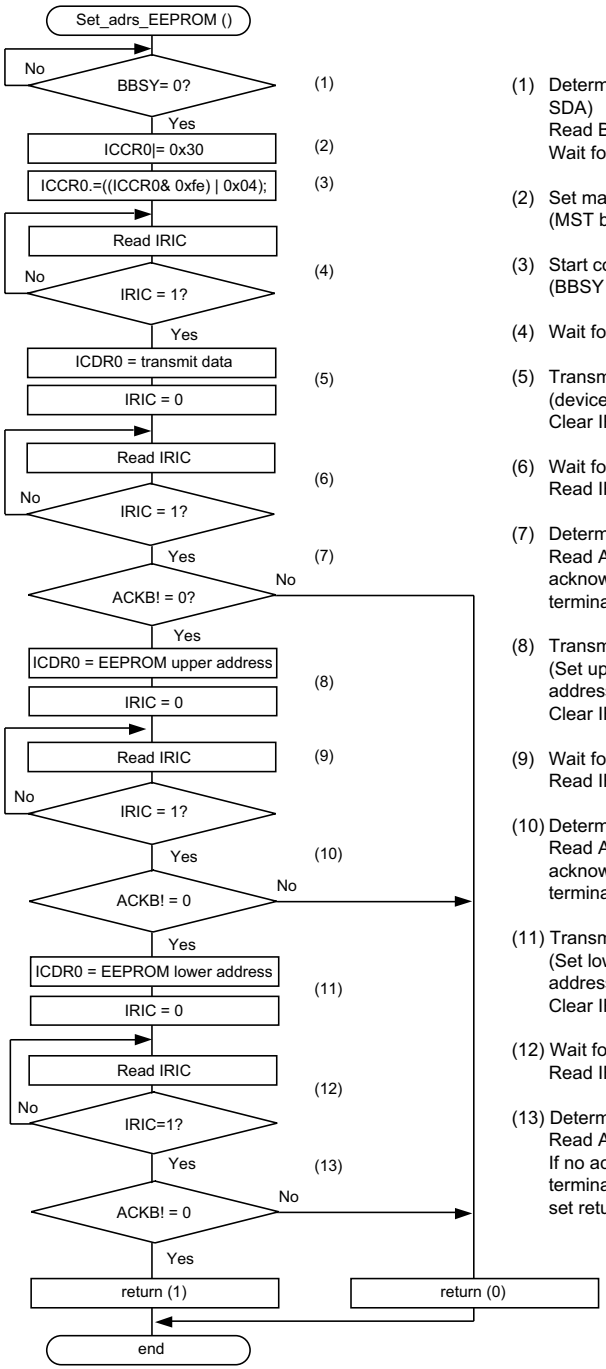
(13) Set transmit data
(device code, slave address, read bits)
Clear IRIC flag bit in ICCR register

(14) Wait for end of transmission of 1 byte
Read IRIC flag bit in ICCR register

(15) Determination of acknowledgment from EEPROM
Read ACKB bit in ICSR register, and if no acknowledgment, set ack variable to 0 and perform end processing (stop condition issuance)


1
ack = 1?
No
(16)
Yes
TRS = 0
WAIT = 0
ACKB = 0
(17)
HNDS = 1
IRIC = 0
1 < num?
No
Yes
Dummy read of ICDR0
(18)
Read IRIC flag
IRIC= 1?
(19)
No
Yes
IRIC = 0
(20)
count = 2
(21)
count < num?
No
Yes
*r_data = ICDR0
(22)
Read IRIC flag
IRIC = 1?
(23)
No
Yes
IRIC = 0
(24)
r_data++
(25)
count++
(26)
ACKB = 1
(27)
num == 1?
No
(28)
Yes
*r_data = ICDR0
Dummy read of ICDR0
r_data++
Read IRIC flag
(29)
IRIC = 1?
No
Yes
IRIC = 0
(30)
TRS = 1
(31)
*r_data= ICDR0
ICCR = ICCR&0xfa
(32)
end
(16) When ack variable = 1, slave address (EEPROM) recognition is successful
Perform data read processing
(17) Set master receive mode
TRS bit = 0 (MST bit = 1)
— No wait (WAIT = 0)
— No continuous reception operation (HNDS = 1)
— Clear IRIC flag bit in ICCR register
(18) Dummy read
(19) Wait for end of reception of 1 byte of data
(20) Clear IRIC flag bit
(21) Set initial value of transfer counter variable
(22) Read receive data
(23) Wait for end of reception of 1 byte of data
(24) Clear IRIC flag bit
(25) Increment receive data storage variable pointer
(26) Increment transfer counter variable
(27) Last reception acknowledge data setting
(28) Receive data read
Dummy read for start of reception when 1st frame of reception is last data
(29) Wait for end of reception of 1 byte of data
(30) Clear IRIC flag bit in ICCR register
(31) Receive data read
(32) Stop condition issuance
(BBSY = 0, SCP = 0)

(4) Start condition issuance, slave address and EEPROM memory address transmission subroutine



(1) Determination of state of I2C Bus lines (SCL, SDA)
Read BBSY flag bit in ICCR register
Wait for bus free state (BBSY = 0)

(2) Set master transmit mode
(MST bit = 1, TRS bit = 1)

(3) Start condition issuance
(BBSY bit = 1, SCP bit = 0)

(4) Wait for start condition generation

(5) Transmit data setting
(device code, slave address, write bits)
Clear IRIC flag bit in ICCR register

(6) Wait for end of transmission of 1 byte
Read IRIC flag bit in ICCR register

(7) Determination of acknowledgment from EEPROM
Read ACKB bit in ICSR register, and if no acknowledgment, set return value to 0 and terminate function

(8) Transmit data setting
(Set upper byte address of EEPROM write start address)
Clear IRIC flag bit in ICCR register

(9) Wait for end of transmission of 1 byte
Read IRIC flag bit in ICCR register

(10) Determination of acknowledgment from EEPROM
Read ACKB bit in ICSR register, and if no acknowledgment, set return value to 0 and terminate function

(11) Transmit data setting
(Set lower byte address of EEPROM write start address)
Clear IRIC flag bit in ICCR register

(12) Wait for end of transmission of 1 byte
Read IRIC flag bit in ICCR register

(13) Determination of acknowledgment from EEPROM
Read ACKB bit in ICSR register
If no acknowledgment, set return value to 0 and terminate function; if acknowledgment present, set return value to 1 and terminate function

2.3.5 Program Listing

```
//*********************************************************************
//   SH7144F Group -SH7145-  I2C-bus Application Note
//         Single master receive
//         n Byte data write/read 64kbit EEPROM
//               Clock :CPU=40MHz  (External input=10MHz)
//                     :Peripheral=40MHz
//         I2c bit rate:156kHz
//         Written     :2003/2/1  Rev.2.0
//*********************************************************************
#include <machine.h>
#include "iodefine.h"


//------------ Symbol Definition ------------------------------------------
#define DEVICE_CODE       0xa0       // EEPROM DEVICE CODE:b'1010
#define SLAVE_ADRS        0x00       // SLAVE ADRS:b'000
#define IIC_DATA_W        0x00       // WRITE DATA:b'0
#define IIC_DATA_R        0x01       // READ DATA:b'1
#define DATA_NUM          10         // data size


//------------ Function Definition ----------------------------------------
void main(void);
void dummy(void);

unsigned char Read_page_EEPROM(unsigned short,unsigned char*,unsigned char);
unsigned char Set_adrs_EEPROM(unsigned short);



//*********************************************************************
//          main
//*********************************************************************
void main(void)
{
      unsigned short    address;              // EEPROM memory address
      unsigned char     read_data[DATA_NUM]; // read data


      address= 0x0000;                        // set EEPROM address


      // EEPROM data read
      Read_page_EEPROM(address,read_data,DATA_NUM);


      while(1);
}
```

```
//*********************************************************
//   dummy interrput function
//*********************************************************
#pragma interrupt(dummy)
void dummy(void)
{
      //   Interrput error
}



//*********************************************************************
//          Read_page_EEPROM
//          argument1   ;read address(unsigned short)
//          argument2   ;read data(unsigned char)
//          argument2   ;read data number (unsigned char)
//          return      ;1=OK/0=NG EEPROM NO_ACK(unsigned char)
//*********************************************************************
unsigned char Read_page_EEPROM(unsigned short adrs,unsigned char* r_data,
unsigned char num)
{
      unsigned char ack;            // ACK flag
      unsigned char count;          // read data number
      unsigned char dummy;          // dummy data

      // Set standby mode
      P_STBY.MSTCR1.BIT.MSTP21 = 0; // disable I2C standby mode

      ack =1;
      P_IIC.SCRX.BIT.IICE = 1;      // Enables CPU access to the register
      P_IIC.ICCR0.BYTE = 0x89;
           // ICE(7)=b'1           Enable I2C bus interface
           // IEIC(6)=b'0          Disables the interrupt
           // MST(5)=b'0           Slave mode
           // TRS(4)=b'0           Receive mode
           // ACKE(3)=b'1          Continuous data transfer is halted
           // BBSY(2)=b'0
           // IRIC(1)=b'0
           // SCP(0)=b'1           Start/stop condition issuance disabling

      // set I2C pin function
      P_PORTB.PBCR1.WORD = 0x0c00;
                           // SDA0(PB3-32pin@SH7145F),SCL0(PB2-31pin@SH7145F)
      P_PORTB.PBCR2.WORD = 0x0000;

      P_IIC.ICMR0.BYTE = 0x38;
           // MILS(7)=b'0          MSB first
           // WAIT(6)=b'0          A wait state is inserted between DATA and ACK
           // CKS2[2:0](5:3)=b'111 Transfer clock select
           //                      156kH@(@P-fai40MHz,IICX=1)
           //                      39.1kH@(@P-fai10MHz,IICX=1)
```

```
P_IIC.SCRX.BYTE = 0x39;
     // IICX(5)=b'1          transfer-rate select,reference CKS bit
     // IICE(4)=b'1          Enables CPU access to the register
     // HNDS(3)=b'1          Set this bit to 1
     // STOPIM(0)=b'1        disables interrupt requests


// Set device code,EEPROM address
ack = Set_adrs_EEPROM(adrs);         // set device code,EEPROM address


if( ack==1){

   P_IIC.ICCR0.BIT.IRIC = 0;          // clear IRIC
   while(P_PORTB.PBDR.BIT.PB2DR!=0);  // check SCL0 pin state == low?


   // Master-Transmission,Generate the start condition.
   P_IIC.ICCR0.BYTE |= 0x30;  // Select master transmit mode(MST=1,TRS=1)
   P_IIC.ICCR0.BYTE=((P_IIC.ICCR0.BYTE & 0xfe)|0x04);
                              // Generate start condition(BBSY=1,SCP=0)
   while( P_IIC.ICCR0.BIT.IRIC==0 );
                              // Wait for a start condition generation


   // Slave address+R Transmission
   P_IIC.ICDR0.BYTE = (unsigned char)(DEVICE_CODE|SLAVE_ADRS|IIC_DATA_R);
                                    // data set
   P_IIC.ICCR0.BIT.IRIC = 0;        // clear IRIC
   while( P_IIC.ICCR0.BIT.IRIC==0 );  // Wait 1byte transmitted
   if( P_IIC.ICSR0.BIT.ACKB!=0 ){     // Test the acknowledge bit
       ack = 0;                       // no ACK
   }
}


if( ack==1 ){
    // Master receive operation (HNDS=1,WAIT=0)
    P_IIC.ICCR0.BIT.TRS = 0;          // Select receive mode(TRS=0)
    P_IIC.ICMR0.BIT.WAIT = 0;         // set wait=0
    P_IIC.ICSR0.BIT.ACKB = 0;         // set ACK data =0
    P_IIC.SCRX.BIT.HNDS = 1;          // set HNDS bit =1

    P_IIC.ICCR0.BIT.IRIC = 0;         // clear IRIC

    // Start data receiving
    if(num>1){                                // case nByte data read (n>1)
        dummy = P_IIC.ICDR0.BYTE;             // dummy read
        while( P_IIC.ICCR0.BIT.IRIC==0 ); // Wait for 1 byte to be received
        P_IIC.ICCR0.BIT.IRIC = 0;             // clear IRIC
        for( count=2; count<num; count++ ){   // (num-2)byte read
```

```
                *r_data = P_IIC.ICDR0.BYTE;  // read receive data
                while( P_IIC.ICCR0.BIT.IRIC==0 );
                                             // Wait for 1 byte to be received
                P_IIC.ICCR0.BIT.IRIC = 0;    // clear IRIC
                r_data++;
            }
        }

        P_IIC.ICSR0.BIT.ACKB = 1;         // set ACK data =1

        if(num==1){                       // case 1Byte read
            dummy = P_IIC.ICDR0.BYTE;     // dummy read
        }else{                            // case nByte data read (n>1)
            *r_data = P_IIC.ICDR0.BYTE;   // read receive data(n-1)
            r_data++;
        }
        while( P_IIC.ICCR0.BIT.IRIC==0 ); // Wait for 1 byte to be received
        P_IIC.ICCR0.BIT.IRIC = 0;         // clear IRIC

        // End data receiving
        P_IIC.ICCR0.BIT.TRS = 1;          // Select transmit mode
        *r_data = P_IIC.ICDR0.BYTE;       // read END receive data
    }

    // Stop condition issuance
    P_IIC.ICCR0.BYTE = P_IIC.ICCR0.BYTE & 0xfa;
                                      // Stop condition issuance(BBSY=0,SCP=0)

    return(ack);
}


//*********************************************************************
//     Set_adrs_EEPROM
//          argument1   ;write address(unsigned short)
//          return      ;1=OK/0=NG EEPROM NO_ACK(unsigned char)
//*********************************************************************
unsigned char Set_adrs_EEPROM(unsigned short adrs)
{
    while( P_IIC.ICCR0.BIT.BBSY!=0 );       // BUS FREE?(BBSY==0→Bus Free)

    // Master-Transmission,Generate the start condition.
    P_IIC.ICCR0.BYTE |= 0x30;       // Select master transmit mode(MST=1,TRS=1)

    P_IIC.ICCR0.BYTE=((P_IIC.ICCR0.BYTE & 0xfe) | 0x04);
                                     // Generate start condition(BBSY=1,SCP=0)
    while( P_IIC.ICCR0.BIT.IRIC==0 ); // Wait for a start condition generation

    // Slave address+W Transmission
    P_IIC.ICDR0.BYTE = (unsigned char)(DEVICE_CODE|SLAVE_ADRS|IIC_DATA_W);
                                                 // data set
```

```
    P_IIC.ICCR0.BIT.IRIC = 0;                    // clear IRIC
    while( P_IIC.ICCR0.BIT.IRIC==0 );            // Wait 1byte transmitted
    if( P_IIC.ICSR0.BIT.ACKB!=0 ){               // Test the acknowledge bit
        return (0);                              // no ACK
    }
    // EEPROM upper address Transmission(1byte)
    P_IIC.ICDR0.BYTE = (unsigned char)(adrs>>8); // data set
    P_IIC.ICCR0.BIT.IRIC = 0;                    // clear IRIC
    while( P_IIC.ICCR0.BIT.IRIC==0 );            // Wait 1byte transmitted

    if( P_IIC.ICSR0.BIT.ACKB!=0 ){               // Test the acknowledge bit
        return (0);                              // no ACK
    }
    // EEPROM lower address Transmission(1byte)
    P_IIC.ICDR0.BYTE = (unsigned char)(adrs & 0x00ff);  // data set
    P_IIC.ICCR0.BIT.IRIC = 0;                    // clear IRIC
    while( P_IIC.ICCR0.BIT.IRIC==0 );            // Wait 1byte transmitted
    if( P_IIC.ICSR0.BIT.ACKB!=0 ){               // Test the acknowledge bit
        return (0);                              // no ACK
    }
    return (1);                                  // ACK OK
}
```

3.1 SH7145F Register Definition File

The SH7145F register definition file is shown below.

```
/************************************************************************/
/*  FILE        :iodefine.h                                             */
/*  DATE        :Tue, Oct 02, 2001                                      */
/*  DESCRIPTION :Definition of I/O Register                             */
/*  CPU TYPE    :SH7145F                                                */
/*  This file is generated by Hitachi Project Generator (Ver.1.2).      */
/************************************************************************/

/************************************************************************/
/*      7145 Include File                                               */
/************************************************************************/
struct st_sci0 {                                    /* struct SCI0  */
       union {                                      /* SMR_0        */
             unsigned char BYTE;                    /*  Byte Access */
             struct {                               /*  Bit Access  */
                    unsigned char CA:1;             /*    C/A       */
                    unsigned char CHR:1;            /*    CHR       */
                    unsigned char PE:1;             /*    PE        */
                    unsigned char OE:1;             /*    O/E       */
                    unsigned char STOP:1;           /*    STOP      */
                    unsigned char MP:1;             /*    MP        */
                    unsigned char CKS:2;            /*    CKS       */
                    } BIT;                          /*              */
             } SMR_0;                               /*              */
       unsigned char BRR_0;                         /* BRR_0        */
       union {                                      /* SCR_0        */
             unsigned char BYTE;                    /*  Byte Access */
             struct {                               /*  Bit Access  */
                    unsigned char TIE:1;            /*    TIE       */
                    unsigned char RIE:1;            /*    RIE       */
                    unsigned char TE:1;             /*    TE        */
                    unsigned char RE:1;             /*    RE        */
                    unsigned char MPIE:1;           /*    MPIE      */
                    unsigned char TEIE:1;           /*    TEIE      */
                    unsigned char CKE:2;            /*    CKE       */
                    } BIT;                          /*              */
             } SCR_0;                               /*              */
       unsigned char TDR_0;                         /* TDR_0        */
       union {                                      /* SSR_0        */
             unsigned char BYTE;                    /*  Byte Access */
             struct {                               /*  Bit Access  */
                    unsigned char TDRE:1;           /*    TDRE      */
                    unsigned char RDRF:1;           /*    RDRF      */
```

```
                    unsigned char ORER:1;             /*    ORER      */
                    unsigned char FER:1;              /*    FER       */
                    unsigned char PER:1;              /*    PER       */
                    unsigned char TEND:1;             /*    TEND      */
                    unsigned char MPB:1;              /*    MPB       */
                    unsigned char MPBT:1;             /*    MPBT      */
                    } BIT;                            /*              */
              } SSR_0;                                /*              */
        unsigned char RDR_0;                          /* RDR_0        */
        union {                                       /* SDCR_0       */
              unsigned char BYTE;                     /*  Byte Access */
              struct {                                /*  Bit Access  */
                    unsigned char :4;                 /*              */
                    unsigned char DIR:1;              /*    DIR       */
                    unsigned char :3;                 /*              */
                    } BIT;                            /*              */
              } SDCR_0;                               /*              */
  };                                                  /*              */
  struct st_sci1 {                                    /* struct SCI1  */
        union {                                       /* SMR_1        */
              unsigned char BYTE;                     /*  Byte Access */
              struct {                                /*  Bit Access  */
                    unsigned char CA:1;               /*    C/A       */
                    unsigned char CHR:1;              /*    CHR       */
                    unsigned char PE:1;               /*    PE        */
                    unsigned char OE:1;               /*    O/E       */
                    unsigned char STOP:1;             /*    STOP      */
                    unsigned char MP:1;               /*    MP        */
                    unsigned char CKS:2;              /*    CKS       */
                    } BIT;                            /*              */
              } SMR_1;                                /*              */
        unsigned char BRR_1;                          /* BRR_1        */
        union {                                       /* SCR_1        */
              unsigned char BYTE;                     /*  Byte Access */
              struct {                                /*  Bit Access  */
                    unsigned char TIE:1;              /*    TIE       */
                    unsigned char RIE:1;              /*    RIE       */
                    unsigned char TE:1;               /*    TE        */
                    unsigned char RE:1;               /*    RE        */
                    unsigned char MPIE:1;             /*    MPIE      */
                    unsigned char TEIE:1;             /*    TEIE      */
                    unsigned char CKE:2;              /*    CKE       */
                    } BIT;                            /*              */
              } SCR_1;                                /*              */
        unsigned char TDR_1;                          /* TDR_1        */
        union {                                       /* SSR_1        */
              unsigned char BYTE;                     /*  Byte Access */
              struct {                                /*  Bit Access  */
                    unsigned char TDRE:1;             /*    TDRE      */
                    unsigned char RDRF:1;             /*    RDRF      */
                    unsigned char ORER:1;             /*    ORER      */
```

```
                  unsigned char FER:1;              /*    FER       */
                  unsigned char PER:1;              /*    PER       */
                  unsigned char TEND:1;             /*    TEND      */
                  unsigned char MPB:1;              /*    MPB       */
                  unsigned char MPBT:1;             /*    MPBT      */
                  } BIT;                            /*              */
            } SSR_1;                                /*              */
      unsigned char RDR_1;                          /* RDR_1        */
      union {                                       /* SDCR_1       */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char :4;                 /*              */
                  unsigned char DIR:1;              /*    DIR       */
                  unsigned char :3;                 /*              */
                  } BIT;                            /*              */
            } SDCR_1;                               /*              */
};                                                  /*              */
struct st_sci2 {                                    /* struct SCI2  */
      union {                                       /* SMR_2        */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char CA:1;               /*    C/A       */
                  unsigned char CHR:1;              /*    CHR       */
                  unsigned char PE:1;               /*    PE        */
                  unsigned char OE:1;               /*    O/E       */
                  unsigned char STOP:1;             /*    STOP      */
                  unsigned char MP:1;               /*    MP        */
                  unsigned char CKS:2;              /*    CKS       */
                  } BIT;                            /*              */
            } SMR_2;                                /*              */
      unsigned char BRR_2;                          /* BRR_2        */
      union {                                       /* SCR_2        */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char TIE:1;              /*    TIE       */
                  unsigned char RIE:1;              /*    RIE       */
                  unsigned char TE:1;               /*    TE        */
                  unsigned char RE:1;               /*    RE        */
                  unsigned char MPIE:1;             /*    MPIE      */
                  unsigned char TEIE:1;             /*    TEIE      */
                  unsigned char CKE:2;              /*    CKE       */
                  } BIT;                            /*              */
            } SCR_2;                                /*              */
      unsigned char TDR_2;                          /* TDR_2        */
      union {                                       /* SSR_2        */
            unsigned char BYTE;                     /*  Byte Access */
            struct {                                /*  Bit Access  */
                  unsigned char TDRE:1;             /*    TDRE      */
                  unsigned char RDRF:1;             /*    RDRF      */
                  unsigned char ORER:1;             /*    ORER      */
                  unsigned char FER:1;              /*    FER       */
```

```
                    unsigned char PER:1;            /*    PER       */
                    unsigned char TEND:1;           /*    TEND      */
                    unsigned char MPB:1;            /*    MPB       */
                    unsigned char MPBT:1;           /*    MPBT      */
                    } BIT;                          /*              */
             } SSR_2;                               /*              */
       unsigned char RDR_2;                         /* RDR_2        */
       union {                                      /* SDCR_2       */
             unsigned char BYTE;                    /*  Byte Access */
             struct {                               /*  Bit Access  */
                    unsigned char :4;               /*              */
                    unsigned char DIR:1;            /*    DIR       */
                    unsigned char :3;               /*              */
                    } BIT;                          /*              */
             } SDCR_2;                              /*              */
};                                                  /*              */
struct st_sci3 {                                    /* struct SCI3  */
       union {                                      /* SMR_3        */
             unsigned char BYTE;                    /*  Byte Access */
             struct {                               /*  Bit Access  */
                    unsigned char CA:1;             /*    C/A       */
                    unsigned char CHR:1;            /*    CHR       */
                    unsigned char PE:1;             /*    PE        */
                    unsigned char OE:1;             /*    O/E       */
                    unsigned char STOP:1;           /*    STOP      */
                    unsigned char MP:1;             /*    MP        */
                    unsigned char CKS:2;            /*    CKS       */
                    } BIT;                          /*              */
             } SMR_3;                               /*              */
       unsigned char BRR_3;                         /* BRR_3        */
       union {                                      /* SCR_3        */
             unsigned char BYTE;                    /*  Byte Access */
             struct {                               /*  Bit Access  */
                    unsigned char TIE:1;            /*    TIE       */
                    unsigned char RIE:1;            /*    RIE       */
                    unsigned char TE:1;             /*    TE        */
                    unsigned char RE:1;             /*    RE        */
                    unsigned char MPIE:1;           /*    MPIE      */
                    unsigned char TEIE:1;           /*    TEIE      */
                    unsigned char CKE:2;            /*    CKE       */
                    } BIT;                          /*              */
             } SCR_3;                               /*              */
       unsigned char TDR_3;                         /* TDR_3        */
       union {                                      /* SSR_3        */
             unsigned char BYTE;                    /*  Byte Access */
             struct {                               /*  Bit Access  */
                    unsigned char TDRE:1;           /*    TDRE      */
                    unsigned char RDRF:1;           /*    RDRF      */
                    unsigned char ORER:1;           /*    ORER      */
                    unsigned char FER:1;            /*    FER       */
                    unsigned char PER:1;            /*    PER       */
```

```
                  unsigned char TEND:1;            /*    TEND      */
                  unsigned char MPB:1;             /*    MPB       */
                  unsigned char MPBT:1;            /*    MPBT      */
                  } BIT;                           /*              */
            } SSR_3;                               /*              */
      unsigned char RDR_3;                         /* RDR_3        */
      union {                                      /* SDCR_3       */
            unsigned char BYTE;                    /*  Byte Access */
            struct {                               /*  Bit Access  */
                  unsigned char :4;                /*              */
                  unsigned char DIR:1;             /*    DIR       */
                  unsigned char :3;                /*              */
                  } BIT;                           /*              */
            } SDCR_3;                              /*              */
};                                                 /*              */
struct st_mtu34 {                                  /* struct MTU34 */
      union {                                      /* TCR_3        */
            unsigned char BYTE;                    /*  Byte Access */
            struct {                               /*  Bit Access  */
                  unsigned char CCLR:3;            /*    CCLR      */
                  unsigned char CKEG:2;            /*    CKEG      */
                  unsigned char TPSC:3;            /*    TPSC      */
                  } BIT;                           /*              */
            } TCR_3;                               /*              */
      union {                                      /* TCR_4        */
            unsigned char BYTE;                    /*  Byte Access */
            struct {                               /*  Bit Access  */
                  unsigned char CCLR:3;            /*    CCLR      */
                  unsigned char CKEG:2;            /*    CKEG      */
                  unsigned char TPSC:3;            /*    TPSC      */
                  } BIT;                           /*              */
            } TCR_4;                               /*              */
      union {                                      /* TMDR_3       */
            unsigned char BYTE;                    /*  Byte Access */
            struct {                               /*  Bit Access  */
                  unsigned char :2;                /*              */
                  unsigned char BFB:1;             /*    BFB       */
                  unsigned char BFA:1;             /*    BFA       */
                  unsigned char MD:4;              /*    MD        */
                  } BIT;                           /*              */
            } TMDR_3;                              /*              */
      union {                                      /* TMDR_4       */
            unsigned char BYTE;                    /*  Byte Access */
            struct {                               /*  Bit Access  */
                  unsigned char :2;                /*              */
                  unsigned char BFB:1;             /*    BFB       */
                  unsigned char BFA:1;             /*    BFA       */
                  unsigned char MD:4;              /*    MD        */
                  } BIT;                           /*              */
            } TMDR_4;                              /*              */
      union {                                      /* TIORH_3      */
```

```
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
               unsigned char IOB:4;             /*    IOB       */
               unsigned char IOA:4;             /*    IOA       */
               } BIT;                           /*              */
        } TIORH_3;                              /*              */
    union {                                     /* TIORL_3      */
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
               unsigned char IOD:4;             /*    IOD       */
               unsigned char IOC:4;             /*    IOC       */
               } BIT;                           /*              */
        } TIORL_3;                              /*              */
    union {                                     /* TIORH_4      */
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
               unsigned char IOB:4;             /*    IOB       */
               unsigned char IOA:4;             /*    IOA       */
               } BIT;                           /*              */
        } TIORH_4;                              /*              */
    union {                                     /* TIORL_4      */
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
               unsigned char IOD:4;             /*    IOD       */
               unsigned char IOC:4;             /*    IOC       */
               } BIT;                           /*              */
        } TIORL_4;                              /*              */
    union {                                     /* TIER_3       */
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
               unsigned char TTGE:1;            /*    TTGE      */
               unsigned char :2;                /*              */
               unsigned char TCIEV:1;           /*    TCIEV     */
               unsigned char TGIED:1;           /*    TGIED     */
               unsigned char TGIEC:1;           /*    TGIEC     */
               unsigned char TGIEB:1;           /*    TGIEB     */
               unsigned char TGIEA:1;           /*    TGIEA     */
               } BIT;                           /*              */
        } TIER_3;                               /*              */
    union {                                     /* TIER_4       */
        unsigned char BYTE;                     /*  Byte Access */
        struct {                                /*  Bit Access  */
               unsigned char TTGE:1;            /*    TTGE      */
               unsigned char :2;                /*              */
               unsigned char TCIEV:1;           /*    TCIEV     */
               unsigned char TGIED:1;           /*    TGIED     */
               unsigned char TGIEC:1;           /*    TGIEC     */
               unsigned char TGIEB:1;           /*    TGIEB     */
               unsigned char TGIEA:1;           /*    TGIEA     */
               } BIT;                           /*              */
        } TIER_4;                               /*              */
```

```
union {                                 /* TOER         */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char :2;          /*              */
             unsigned char OE4D:1;      /*    OE4D      */
             unsigned char OE4C:1;      /*    OE4C      */
             unsigned char OE3D:1;      /*    OE3D      */
             unsigned char OE4B:1;      /*    OE4B      */
             unsigned char OE4A:1;      /*    OE4A      */
             unsigned char OE3B:1;      /*    OE3B      */
             } BIT;                     /*              */
      } TOER;                           /*              */
union {                                 /* TOCR         */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char :1;          /*              */
             unsigned char PSYE:1;      /*    PSYE      */
             unsigned char :4;          /*              */
             unsigned char OLSN:1;      /*    OLSN      */
             unsigned char OLSP:1;      /*    OLSP      */
             } BIT;                     /*              */
      } TOCR;                           /*              */
unsigned char wk0[1];                   /*              */
union {                                 /* TGCR         */
      unsigned char BYTE;               /*  Byte Access */
      struct {                          /*  Bit Access  */
             unsigned char :1;          /*              */
             unsigned char BDC:1;       /*    BDC       */
             unsigned char N:1;         /*    N         */
             unsigned char P:1;         /*    P         */
             unsigned char FB:1;        /*    FB        */
             unsigned char WF:1;        /*    WF        */
             unsigned char VF:1;        /*    VF        */
             unsigned char UF:1;        /*    UF        */
             } BIT;                     /*              */
      } TGCR;                           /*              */
unsigned char wk1[2];                   /*              */
unsigned short TCNT_3;                  /* TCNT_3       */
unsigned short TCNT_4;                  /* TCNT_4       */
unsigned short TCDR;                    /* TCDR         */
unsigned short TDDR;                    /* TDDR         */
unsigned short TGRA_3;                  /* TGRA_3       */
unsigned short TGRB_3;                  /* TGRB_3       */
unsigned short TGRA_4;                  /* TGRA_4       */
unsigned short TGRB_4;                  /* TGRB_4       */
unsigned short TCNTS;                   /* TCNTS        */
unsigned short TCBR;                    /* TCBR         */
unsigned short TGRC_3;                  /* TGRC_3       */
unsigned short TGRD_3;                  /* TGRD_3       */
unsigned short TGRC_4;                  /* TGRC_4       */
unsigned short TGRD_4;                  /* TGRD_4       */
```

```
        union {                                 /* TSR_3        */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                    unsigned char TCFD:1;       /*    TCFD      */
                    unsigned char :2;           /*              */
                    unsigned char TCFV:1;       /*    TCFV      */
                    unsigned char TGFD:1;       /*    TGFD      */
                    unsigned char TGFC:1;       /*    TGFC      */
                    unsigned char TGFB:1;       /*    TGFB      */
                    unsigned char TGFA:1;       /*    TGFA      */
                    } BIT;                      /*              */
              } TSR_3;                          /*              */
        union {                                 /* TSR_4        */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                    unsigned char TCFD:1;       /*    TCFD      */
                    unsigned char :2;           /*              */
                    unsigned char TCFV:1;       /*    TCFV      */
                    unsigned char TGFD:1;       /*    TGFD      */
                    unsigned char TGFC:1;       /*    TGFC      */
                    unsigned char TGFB:1;       /*    TGFB      */
                    unsigned char TGFA:1;       /*    TGFA      */
                    } BIT;                      /*              */
              } TSR_4;                          /*              */
        unsigned char wk2[18];                  /*              */
        union {                                 /* TSTR         */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                    unsigned char CST4:1;       /*    CST4      */
                    unsigned char CST3:1;       /*    CST3      */
                    unsigned char :3;           /*              */
                    unsigned char CST:3;        /*    CST       */
                    } BIT;                      /*              */
              } TSTR;                           /*              */
        union {                                 /* TSYR         */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                    unsigned char SYNC4:1;      /*    SYNC4     */
                    unsigned char SYNC3:1;      /*    SYNC3     */
                    unsigned char :3;           /*              */
                    unsigned char SYNC:3;       /*    SYNC      */
                    } BIT;                      /*              */
              } TSYR;                           /*              */
  };                                            /*              */
  struct st_mtu0 {                              /* struct MTU0  */
        union {                                 /* TCR_0        */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                    unsigned char CCLR:3;       /*    CCLR      */
                    unsigned char CKEG:2;       /*    CKEG      */
                    unsigned char TPSC:3;       /*    TPSC      */
```

```
            } BIT;                              /*              */
      } TCR_0;                                  /*              */
union {                                         /* TMDR_0       */
      unsigned char BYTE;                       /*  Byte Access */
      struct {                                  /*  Bit Access  */
            unsigned char :2;                   /*              */
            unsigned char BFB:1;                /*    BFB       */
            unsigned char BFA:1;                /*    BFA       */
            unsigned char MD:4;                 /*    MD        */
            } BIT;                              /*              */
      } TMDR_0;                                 /*              */
union {                                         /* TIORH_0      */
      unsigned char BYTE;                       /*  Byte Access */
      struct {                                  /*  Bit Access  */
            unsigned char IOB:4;                /*    IOB       */
            unsigned char IOA:4;                /*    IOA       */
            } BIT;                              /*              */
      } TIORH_0;                                /*              */
union {                                         /* TIORL_0      */
      unsigned char BYTE;                       /*  Byte Access */
      struct {                                  /*  Bit Access  */
            unsigned char IOD:4;                /*    IOD       */
            unsigned char IOC:4;                /*    IOC       */
            } BIT;                              /*              */
      } TIORL_0;                                /*              */
union {                                         /* TIER_0       */
      unsigned char BYTE;                       /*  Byte Access */
      struct {                                  /*  Bit Access  */
            unsigned char TTGE:1;               /*    TTGE      */
            unsigned char :2;                   /*              */
            unsigned char TCIEV:1;              /*    TCIEV     */
            unsigned char TGIED:1;              /*    TGIED     */
            unsigned char TGIEC:1;              /*    TGIEC     */
            unsigned char TGIEB:1;              /*    TGIEB     */
            unsigned char TGIEA:1;              /*    TGIEA     */
            } BIT;                              /*              */
      } TIER_0;                                 /*              */
union {                                         /* TSR_0        */
      unsigned char BYTE;                       /*  Byte Access */
      struct {                                  /*  Bit Access  */
            unsigned char :3;                   /*              */
            unsigned char TCFV:1;               /*    TCFV      */
            unsigned char TGFD:1;               /*    TGFD      */
            unsigned char TGFC:1;               /*    TGFC      */
            unsigned char TGFB:1;               /*    TGFB      */
            unsigned char TGFA:1;               /*    TGFA      */
            } BIT;                              /*              */
      } TSR_0;                                  /*              */
unsigned short TCNT_0;                          /* TCNT_0       */
unsigned short TGRA_0;                          /* TGRA_0       */
unsigned short TGRB_0;                          /* TGRB_0       */
```

```
        unsigned short TGRC_0;                  /* TGRC_0       */
        unsigned short TGRD_0;                  /* TGRD_0       */
};                                              /*              */
struct st_mtu1 {                                /* struct MTU1  */
        union {                                 /* TCR_1        */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char :1;          /*              */
                     unsigned char CCLR:2;      /*    CCLR      */
                     unsigned char CKEG:2;      /*    CKEG      */
                     unsigned char TPSC:3;      /*    TPSC      */
                     } BIT;                     /*              */
              } TCR_1;                          /*              */
        union {                                 /* TMDR_1       */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char :4;          /*              */
                     unsigned char MD:4;        /*    MD        */
                     } BIT;                     /*              */
              } TMDR_1;                         /*              */
        union {                                 /* TIOR_1       */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char IOB:4;       /*    IOB       */
                     unsigned char IOA:4;       /*    IOA       */
                     } BIT;                     /*              */
              } TIOR_1;                         /*              */
        unsigned char wk0[1];                   /*              */
        union {                                 /* TIER_1       */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char TTGE:1;      /*    TTGE      */
                     unsigned char :1;          /*              */
                     unsigned char TCIEU:1;     /*    TCIEU     */
                     unsigned char TCIEV:1;     /*    TCIEV     */
                     unsigned char :2;          /*              */
                     unsigned char TGIEB:1;     /*    TGIEB     */
                     unsigned char TGIEA:1;     /*    TGIEA     */
                     } BIT;                     /*              */
              } TIER_1;                         /*              */
        union {                                 /* TSR_1        */
              unsigned char BYTE;               /*  Byte Access */
              struct {                          /*  Bit Access  */
                     unsigned char TCFD:1;      /*    TCFD      */
                     unsigned char :1;          /*              */
                     unsigned char TCFU:1;      /*    TCFU      */
                     unsigned char TCFV:1;      /*    TCFV      */
                     unsigned char :2;          /*              */
                     unsigned char TGFB:1;      /*    TGFB      */
                     unsigned char TGFA:1;      /*    TGFA      */
                     } BIT;                     /*              */
```

```
            } TSR_1;                                    /*              */
        unsigned short TCNT_1;                          /* TCNT_1       */
        unsigned short TGRA_1;                          /* TGRA_1       */
        unsigned short TGRB_1;                          /* TGRB_1       */
};                                                      /*              */
struct st_mtu2 {                                        /* struct MTU2  */
        union {                                         /* TCR_2        */
            unsigned char BYTE;                         /*  Byte Access */
            struct {                                    /*  Bit Access  */
                unsigned char :1;                       /*              */
                unsigned char CCLR:2;                   /*    CCLR      */
                unsigned char CKEG:2;                   /*    CKEG      */
                unsigned char TPSC:3;                   /*    TPSC      */
                } BIT;                                  /*              */
            } TCR_2;                                    /*              */
        union {                                         /* TMDR_2       */
            unsigned char BYTE;                         /*  Byte Access */
            struct {                                    /*  Bit Access  */
                unsigned char :4;                       /*              */
                unsigned char MD:4;                     /*    MD        */
                } BIT;                                  /*              */
            } TMDR_2;                                   /*              */
        union {                                         /* TIOR_2       */
            unsigned char BYTE;                         /*  Byte Access */
            struct {                                    /*  Bit Access  */
                unsigned char IOB:4;                    /*    IOB       */
                unsigned char IOA:4;                    /*    IOA       */
                } BIT;                                  /*              */
            } TIOR_2;                                   /*              */
        unsigned char wk0[1];                           /*              */
        union {                                         /* TIER_2       */
            unsigned char BYTE;                         /*  Byte Access */
            struct {                                    /*  Bit Access  */
                unsigned char TTGE:1;                   /*    TTGE      */
                unsigned char :1;                       /*              */
                unsigned char TCIEU:1;                  /*    TCIEU     */
                unsigned char TCIEV:1;                  /*    TCIEV     */
                unsigned char :2;                       /*              */
                unsigned char TGIEB:1;                  /*    TGIEB     */
                unsigned char TGIEA:1;                  /*    TGIEA     */
                } BIT;                                  /*              */
            } TIER_2;                                   /*              */
        union {                                         /* TSR_2        */
            unsigned char BYTE;                         /*  Byte Access */
            struct {                                    /*  Bit Access  */
                unsigned char TCFD:1;                   /*    TCFD      */
                unsigned char :1;                       /*              */
                unsigned char TCFU:1;                   /*    TCFU      */
                unsigned char TCFV:1;                   /*    TCFV      */
                unsigned char :2;                       /*              */
                unsigned char TGFB:1;                   /*    TGFB      */
```

```
                    unsigned char TGFA:1;           /*    TGFA      */
                    } BIT;                          /*              */
              } TSR_2;                              /*              */
        unsigned short TCNT_2;                      /* TCNT_2       */
        unsigned short TGRA_2;                      /* TGRA_2       */
        unsigned short TGRB_2;                      /* TGRB_2       */
  };                                                /*              */
  struct st_intc {                                  /* struct INTC  */
        union {                                     /* IPRA         */
              unsigned short WORD;                  /*  Word Access */
              struct {                              /*  Bit Access  */
                    unsigned short IRQ0:4;          /*    IRQ0      */
                    unsigned short IRQ1:4;          /*    IRQ1      */
                    unsigned short IRQ2:4;          /*    IRQ2      */
                    unsigned short IRQ3:4;          /*    IRQ3      */
                    } BIT;                          /*              */
              } IPRA;                               /*              */
        union {                                     /* IPRB         */
              unsigned short WORD;                  /*  Word Access */
              struct {                              /*  Bit Access  */
                    unsigned short IRQ4:4;          /*    IRQ4      */
                    unsigned short IRQ5:4;          /*    IRQ5      */
                    unsigned short IRQ6:4;          /*    IRQ6      */
                    unsigned short IRQ7:4;          /*    IRQ7      */
                    } BIT;                          /*              */
              } IPRB;                               /*              */
        union {                                     /* IPRC         */
              unsigned short WORD;                  /*  Word Access */
              struct {                              /*  Bit Access  */
                    unsigned short DMAC0:4;         /*    DMAC0     */
                    unsigned short DMAC1:4;         /*    DMAC1     */
                    unsigned short DMAC2:4;         /*    DMAC2     */
                    unsigned short DMAC3:4;         /*    DMAC3     */
                    } BIT;                          /*              */
              } IPRC;                               /*              */
        union {                                     /* IPRD         */
              unsigned short WORD;                  /*  Word Access */
              struct {                              /*  Bit Access  */
                    unsigned short MTU0:8;          /*    MTU0      */
                    unsigned short MTU1:8;          /*    MTU1      */
                    } BIT;                          /*              */
              } IPRD;                               /*              */
        union {                                     /* IPRE         */
              unsigned short WORD;                  /*  Word Access */
              struct {                              /*  Bit Access  */
                    unsigned short MTU2:8;          /*    MTU2      */
                    unsigned short MTU3:8;          /*    MTU3      */
                    } BIT;                          /*              */
              } IPRE;                               /*              */
        union {                                     /* IPRF         */
              unsigned short WORD;                  /*  Word Access */
```

```
      struct {                                  /*  Bit Access  */
             unsigned short MTU4:8;             /*    MTU4      */
             unsigned short SCI0:4;             /*    SCI0      */
             unsigned short SCI1:4;             /*    SCI1      */
             } BIT;                             /*              */
      } IPRF;                                   /*              */
union {                                         /* IPRG         */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
             unsigned short AD01:4;             /*    A/D0,1    */
             unsigned short DTC:4;              /*    DTC       */
             unsigned short CMT0:4;             /*    CMT0      */
             unsigned short CMT1:4;             /*    CMT1      */
             } BIT;                             /*              */
      } IPRG;                                   /*              */
union {                                         /* IPRH         */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
             unsigned short WDT:4;              /*    WDT       */
             unsigned short IOMTU:4;            /*    I/O(MTU)  */
             unsigned short :8;                 /*              */
             } BIT;                             /*              */
      } IPRH;                                   /*              */
union {                                         /* ICR1         */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
             unsigned short NMIL:1;             /*    NMIL      */
             unsigned short :6;                 /*              */
             unsigned short NMIE:1;             /*    NMIE      */
             unsigned short IRQ0S:1;            /*    IRQ0S     */
             unsigned short IRQ1S:1;            /*    IRQ1S     */
             unsigned short IRQ2S:1;            /*    IRQ2S     */
             unsigned short IRQ3S:1;            /*    IRQ3S     */
             unsigned short IRQ4S:1;            /*    IRQ4S     */
             unsigned short IRQ5S:1;            /*    IRQ5S     */
             unsigned short IRQ6S:1;            /*    IRQ6S     */
             unsigned short IRQ7S:1;            /*    IRQ7S     */
             } BIT;                             /*              */
      } ICR1;                                   /*              */
union {                                         /* ISR          */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
             unsigned short :8;                 /*              */
             unsigned short IRQ0F:1;            /*    IRQ0F     */
             unsigned short IRQ1F:1;            /*    IRQ1F     */
             unsigned short IRQ2F:1;            /*    IRQ2F     */
             unsigned short IRQ3F:1;            /*    IRQ3F     */
             unsigned short IRQ4F:1;            /*    IRQ4F     */
             unsigned short IRQ5F:1;            /*    IRQ5F     */
             unsigned short IRQ6F:1;            /*    IRQ6F     */
             unsigned short IRQ7F:1;            /*    IRQ7F     */
```

```
                } BIT;                                  /*              */
          } ISR;                                        /*              */
          union {                                       /* IPRI         */
                unsigned short WORD;                    /*  Word Access */
                struct {                                /*  Bit Access  */
                      unsigned short SCI2:4;            /*    SCI2      */
                      unsigned short SCI3:4;            /*    SCI3      */
                      unsigned short :8;                /*              */
                      } BIT;                            /*              */
          } IPRI;                                       /*              */
          union {                                       /* IPRJ         */
                unsigned short WORD;                    /*  Word Access */
                struct {                                /*  Bit Access  */
                      unsigned short :8;                /*              */
                      unsigned short IIC:4;             /*    IIC       */
                      unsigned short :4;                /*              */
                      } BIT;                            /*              */
          } IPRJ;                                       /*              */
          unsigned char wk0[6];                         /*              */
          union {                                       /* ICR2         */
                unsigned short WORD;                    /*  Word Access */
                struct {                                /*  Bit Access  */
                      unsigned short IRQ0ES:2;          /*    IRQ0ES    */
                      unsigned short IRQ1ES:2;          /*    IRQ1ES    */
                      unsigned short IRQ2ES:2;          /*    IRQ2ES    */
                      unsigned short IRQ3ES:2;          /*    IRQ3ES    */
                      unsigned short IRQ4ES:2;          /*    IRQ4ES    */
                      unsigned short IRQ5ES:2;          /*    IRQ5ES    */
                      unsigned short IRQ6ES:2;          /*    IRQ6ES    */
                      unsigned short IRQ7ES:2;          /*    IRQ7ES    */
                      } BIT;                            /*              */
                } ICR2;                                 /*              */
    };                                                  /*              */
    struct st_porta {                                   /* struct PORTA */
          union {                                       /* PADRH        */
                unsigned short WORD;                    /*  Word Access */
                struct {                                /*  Bit Access  */
                      unsigned short :8;                /*              */
                      unsigned short PA23DR:1;          /*    PA23DR    */
                      unsigned short PA22DR:1;          /*    PA22DR    */
                      unsigned short PA21DR:1;          /*    PA21DR    */
                      unsigned short PA20DR:1;          /*    PA20DR    */
                      unsigned short PA19DR:1;          /*    PA19DR    */
                      unsigned short PA18DR:1;          /*    PA18DR    */
                      unsigned short PA17DR:1;          /*    PA17DR    */
                      unsigned short PA16DR:1;          /*    PA16DR    */
                      } BIT;                            /*              */
                } PADRH;                                /*              */
          union {                                       /* PADRL        */
                unsigned short WORD;                    /*  Word Access */
                struct {                                /*  Bit Access  */
```

```
            unsigned short PA15DR:1;          /*    PA15DR    */
            unsigned short PA14DR:1;          /*    PA14DR    */
            unsigned short PA13DR:1;          /*    PA13DR    */
            unsigned short PA12DR:1;          /*    PA12DR    */
            unsigned short PA11DR:1;          /*    PA11DR    */
            unsigned short PA10DR:1;          /*    PA10DR    */
            unsigned short PA9DR:1;           /*    PA9DR     */
            unsigned short PA8DR:1;           /*    PA8DR     */
            unsigned short PA7DR:1;           /*    PA7DR     */
            unsigned short PA6DR:1;           /*    PA6DR     */
            unsigned short PA5DR:1;           /*    PA5DR     */
            unsigned short PA4DR:1;           /*    PA4DR     */
            unsigned short PA3DR:1;           /*    PA3DR     */
            unsigned short PA2DR:1;           /*    PA2DR     */
            unsigned short PA1DR:1;           /*    PA1DR     */
            unsigned short PA0DR:1;           /*    PA0DR     */
            } BIT;                            /*              */
      } PADRL;                                /*              */
union {                                       /* PAIORH       */
      unsigned short WORD;                    /*  Word Access */
      struct {                                /*  Bit Access  */
            unsigned short :8;                /*              */
            unsigned short PA23IOR:1;         /*    PA23IOR   */
            unsigned short PA22IOR:1;         /*    PA22IOR   */
            unsigned short PA21IOR:1;         /*    PA21IOR   */
            unsigned short PA20IOR:1;         /*    PA20IOR   */
            unsigned short PA19IOR:1;         /*    PA19IOR   */
            unsigned short PA18IOR:1;         /*    PA18IOR   */
            unsigned short PA17IOR:1;         /*    PA17IOR   */
            unsigned short PA16IOR:1;         /*    PA16IOR   */
            } BIT;                            /*              */
      } PAIORH;                               /*              */
union {                                       /* PAIORL       */
      unsigned short WORD;                    /*  Word Access */
      struct {                                /*  Bit Access  */
            unsigned short PA15IOR:1;         /*    PA15IOR   */
            unsigned short PA14IOR:1;         /*    PA14IOR   */
            unsigned short PA13IOR:1;         /*    PA13IOR   */
            unsigned short PA12IOR:1;         /*    PA12IOR   */
            unsigned short PA11IOR:1;         /*    PA11IOR   */
            unsigned short PA10IOR:1;         /*    PA10IOR   */
            unsigned short PA9IOR:1;          /*    PA9IOR    */
            unsigned short PA8IOR:1;          /*    PA8IOR    */
            unsigned short PA7IOR:1;          /*    PA7IOR    */
            unsigned short PA6IOR:1;          /*    PA6IOR    */
            unsigned short PA5IOR:1;          /*    PA5IOR    */
            unsigned short PA4IOR:1;          /*    PA4IOR    */
            unsigned short PA3IOR:1;          /*    PA3IOR    */
            unsigned short PA2IOR:1;          /*    PA2IOR    */
            unsigned short PA1IOR:1;          /*    PA1IOR    */
            unsigned short PA0IOR:1;          /*    PA0IOR    */
```

```
                } BIT;                          /*              */
        } PAIORL;                               /*              */
        union {                                 /* PACRH        */
                unsigned short WORD;            /*  Word Access */
                struct {                        /*  Bit Access  */
                        unsigned short :1;      /*              */
                        unsigned short PA23MD:1;/*    PA23MD    */
                        unsigned short :1;      /*              */
                        unsigned short PA22MD:1;/*    PA22MD    */
                        unsigned short :1;      /*              */
                        unsigned short PA21MD:1;/*    PA21MD    */
                        unsigned short :1;      /*              */
                        unsigned short PA20MD:1;/*    PA20MD    */
                        unsigned short PA19MD:2;/*    PA19MD    */
                        unsigned short PA18MD:2;/*    PA18MD    */
                        unsigned short PA17MD:2;/*    PA17MD    */
                        unsigned short PA16MD:2;/*    PA16MD    */
                        } BIT;                  /*              */
                } PACRH;                        /*              */
        unsigned char wk0[2];                   /*              */
        union {                                 /* PACRL1       */
                unsigned short WORD;            /*  Word Access */
                struct {                        /*  Bit Access  */
                        unsigned short PA15MD:2;/*    PA15MD    */
                        unsigned short PA14MD:2;/*    PA14MD    */
                        unsigned short PA13MD:2;/*    PA13MD    */
                        unsigned short PA12MD:2;/*    PA12MD    */
                        unsigned short PA11MD:2;/*    PA11MD    */
                        unsigned short PA10MD:2;/*    PA10MD    */
                        unsigned short PA9MD:2; /*    PA9MD     */
                        unsigned short PA8MD:2; /*    PA8MD     */
                        } BIT;                  /*              */
                } PACRL1;                       /*              */
        union {                                 /* PACRL2       */
                unsigned short WORD;            /*  Word Access */
                struct {                        /*  Bit Access  */
                        unsigned short PA7MD:2; /*    PA7MD     */
                        unsigned short PA6MD:2; /*    PA6MD     */
                        unsigned short PA5MD:2; /*    PA5MD     */
                        unsigned short PA4MD:2; /*    PA4MD     */
                        unsigned short PA3MD:2; /*    PA3MD     */
                        unsigned short PA2MD:2; /*    PA2MD     */
                        unsigned short PA1MD:2; /*    PA1MD     */
                        unsigned short PA0MD:2; /*    PA0MD     */
                        } BIT;                  /*              */
                } PACRL2;                       /*              */
};                                              /*              */
struct st_portb {                               /* struct PORTB */
        union {                                 /* PBDR         */
                unsigned short WORD;            /*  Word Access */
                struct {                        /*  Bit Access  */
```

```
            unsigned short :6;              /*              */
            unsigned short PB9DR:1;         /*    PB9DR     */
            unsigned short PB8DR:1;         /*    PB8DR     */
            unsigned short PB7DR:1;         /*    PB7DR     */
            unsigned short PB6DR:1;         /*    PB6DR     */
            unsigned short PB5DR:1;         /*    PB5DR     */
            unsigned short PB4DR:1;         /*    PB4DR     */
            unsigned short PB3DR:1;         /*    PB3DR     */
            unsigned short PB2DR:1;         /*    PB2DR     */
            unsigned short PB1DR:1;         /*    PB1DR     */
            unsigned short PB0DR:1;         /*    PB0DR     */
            } BIT;                          /*              */
      } PBDR;                               /*              */
unsigned char wk0[2];                       /*              */
union {                                     /* PBIOR        */
      unsigned short WORD;                  /*  Word Access */
      struct {                              /*  Bit Access  */
            unsigned short :6;              /*              */
            unsigned short PB9IOR:1;        /*    PB9IOR    */
            unsigned short PB8IOR:1;        /*    PB8IOR    */
            unsigned short PB7IOR:1;        /*    PB7IOR    */
            unsigned short PB6IOR:1;        /*    PB6IOR    */
            unsigned short PB5IOR:1;        /*    PB5IOR    */
            unsigned short PB4IOR:1;        /*    PB4IOR    */
            unsigned short PB3IOR:1;        /*    PB3IOR    */
            unsigned short PB2IOR:1;        /*    PB2IOR    */
            unsigned short PB1IOR:1;        /*    PB1IOR    */
            unsigned short PB0IOR:1;        /*    PB0IOR    */
            } BIT;                          /*              */
      } PBIOR;                              /*              */
unsigned char wk1[2];                       /*              */
union {                                     /* PBCR1        */
      unsigned short WORD;                  /*  Word Access */
      struct {                              /*  Bit Access  */
            unsigned short :4;              /*              */
            unsigned short PB3MD2:1;        /*    PB3MD2    */
            unsigned short PB2MD2:1;        /*    PB2MD2    */
            unsigned short :6;              /*              */
            unsigned short PB9MD:2;         /*    PB9MD     */
            unsigned short PB8MD:2;         /*    PB8MD     */
            } BIT;                          /*              */
      } PBCR1;                              /*              */
union {                                     /* PBCR2        */
      unsigned short WORD;                  /*  Word Access */
      struct {                              /*  Bit Access  */
            unsigned short PB7MD:2;         /*    PB7MD     */
            unsigned short PB6MD:2;         /*    PB6MD     */
            unsigned short PB5MD:2;         /*    PB5MD     */
            unsigned short PB4MD:2;         /*    PB4MD     */
            unsigned short PB3MD:2;         /*    PB3MD     */
            unsigned short PB2MD:2;         /*    PB2MD     */
```

```
                    unsigned short PB1MD:2;          /*    PB1MD     */
                    unsigned short PB0MD:2;          /*    PB0MD     */
                    } BIT;                           /*              */
              } PBCR2;                               /*              */
};                                                   /*              */
struct st_portc {                                    /* struct PORTC */
       union {                                       /* PCDR         */
             unsigned short WORD;                    /*  Word Access */
             struct {                                /*  Bit Access  */
                    unsigned short PC15DR:1;         /*    PC15DR    */
                    unsigned short PC14DR:1;         /*    PC14DR    */
                    unsigned short PC13DR:1;         /*    PC13DR    */
                    unsigned short PC12DR:1;         /*    PC12DR    */
                    unsigned short PC11DR:1;         /*    PC11DR    */
                    unsigned short PC10DR:1;         /*    PC10DR    */
                    unsigned short PC9DR:1;          /*    PC9DR     */
                    unsigned short PC8DR:1;          /*    PC8DR     */
                    unsigned short PC7DR:1;          /*    PC7DR     */
                    unsigned short PC6DR:1;          /*    PC6DR     */
                    unsigned short PC5DR:1;          /*    PC5DR     */
                    unsigned short PC4DR:1;          /*    PC4DR     */
                    unsigned short PC3DR:1;          /*    PC3DR     */
                    unsigned short PC2DR:1;          /*    PC2DR     */
                    unsigned short PC1DR:1;          /*    PC1DR     */
                    unsigned short PC0DR:1;          /*    PC0DR     */
                    } BIT;                           /*              */
             } PCDR;                                 /*              */
       unsigned char wk0[2];                         /*              */
       union {                                       /* PCIOR        */
             unsigned short WORD;                    /*  Word Access */
             struct {                                /*  Bit Access  */
                    unsigned short PC15IOR:1;        /*    PC15IOR   */
                    unsigned short PC14IOR:1;        /*    PC14IOR   */
                    unsigned short PC13IOR:1;        /*    PC13IOR   */
                    unsigned short PC12IOR:1;        /*    PC12IOR   */
                    unsigned short PC11IOR:1;        /*    PC11IOR   */
                    unsigned short PC10IOR:1;        /*    PC10IOR   */
                    unsigned short PC9IOR:1;         /*    PC9IOR    */
                    unsigned short PC8IOR:1;         /*    PC8IOR    */
                    unsigned short PC7IOR:1;         /*    PC7IOR    */
                    unsigned short PC6IOR:1;         /*    PC6IOR    */
                    unsigned short PC5IOR:1;         /*    PC5IOR    */
                    unsigned short PC4IOR:1;         /*    PC4IOR    */
                    unsigned short PC3IOR:1;         /*    PC3IOR    */
                    unsigned short PC2IOR:1;         /*    PC2IOR    */
                    unsigned short PC1IOR:1;         /*    PC1IOR    */
                    unsigned short PC0IOR:1;         /*    PC0IOR    */
                    } BIT;                           /*              */
             } PCIOR;                                /*              */
       unsigned char wk1[4];                         /*              */
       union {                                       /* PCCR         */
```

```
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                  unsigned short PC15MD:1;          /*    PC15MD    */
                  unsigned short PC14MD:1;          /*    PC14MD    */
                  unsigned short PC13MD:1;          /*    PC13MD    */
                  unsigned short PC12MD:1;          /*    PC12MD    */
                  unsigned short PC11MD:1;          /*    PC11MD    */
                  unsigned short PC10MD:1;          /*    PC10MD    */
                  unsigned short PC9MD:1;           /*    PC9MD     */
                  unsigned short PC8MD:1;           /*    PC8MD     */
                  unsigned short PC7MD:1;           /*    PC7MD     */
                  unsigned short PC6MD:1;           /*    PC6MD     */
                  unsigned short PC5MD:1;           /*    PC5MD     */
                  unsigned short PC4MD:1;           /*    PC4MD     */
                  unsigned short PC3MD:1;           /*    PC3MD     */
                  unsigned short PC2MD:1;           /*    PC2MD     */
                  unsigned short PC1MD:1;           /*    PC1MD     */
                  unsigned short PC0MD:1;           /*    PC0MD     */
                  } BIT;                            /*              */
            } PCCR;                                 /*              */
};                                                  /*              */
struct st_portd {                                   /* struct PORTD */
      union {                                       /* PDDRH        */
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                  unsigned short PD31DR:1;          /*    PD31DR    */
                  unsigned short PD30DR:1;          /*    PD30DR    */
                  unsigned short PD29DR:1;          /*    PD29DR    */
                  unsigned short PD28DR:1;          /*    PD28DR    */
                  unsigned short PD27DR:1;          /*    PD27DR    */
                  unsigned short PD26DR:1;          /*    PD26DR    */
                  unsigned short PD25DR:1;          /*    PD25DR    */
                  unsigned short PD24DR:1;          /*    PD24DR    */
                  unsigned short PD23DR:1;          /*    PD23DR    */
                  unsigned short PD22DR:1;          /*    PD22DR    */
                  unsigned short PD21DR:1;          /*    PD21DR    */
                  unsigned short PD20DR:1;          /*    PD20DR    */
                  unsigned short PD19DR:1;          /*    PD19DR    */
                  unsigned short PD18DR:1;          /*    PD18DR    */
                  unsigned short PD17DR:1;          /*    PD17DR    */
                  unsigned short PD16DR:1;          /*    PD16DR    */
                  } BIT;                            /*              */
            } PDDRH;                                /*              */
      union {                                       /* PDDRL        */
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                  unsigned short PD15DR:1;          /*    PD15DR    */
                  unsigned short PD14DR:1;          /*    PD14DR    */
                  unsigned short PD13DR:1;          /*    PD13DR    */
                  unsigned short PD12DR:1;          /*    PD12DR    */
                  unsigned short PD11DR:1;          /*    PD11DR    */
```

```
                unsigned short PD10DR:1;          /*    PD10DR    */
                unsigned short PD9DR:1;           /*    PD9DR     */
                unsigned short PD8DR:1;           /*    PD8DR     */
                unsigned short PD7DR:1;           /*    PD7DR     */
                unsigned short PD6DR:1;           /*    PD6DR     */
                unsigned short PD5DR:1;           /*    PD5DR     */
                unsigned short PD4DR:1;           /*    PD4DR     */
                unsigned short PD3DR:1;           /*    PD3DR     */
                unsigned short PD2DR:1;           /*    PD2DR     */
                unsigned short PD1DR:1;           /*    PD1DR     */
                unsigned short PD0DR:1;           /*    PD0DR     */
                } BIT;                            /*              */
        } PDDRL;                                  /*              */
    union {                                       /* PDIORH       */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short PD31IOR:1;         /*    PD31IOR   */
                unsigned short PD30IOR:1;         /*    PD30IOR   */
                unsigned short PD29IOR:1;         /*    PD29IOR   */
                unsigned short PD28IOR:1;         /*    PD28IOR   */
                unsigned short PD27IOR:1;         /*    PD27IOR   */
                unsigned short PD26IOR:1;         /*    PD26IOR   */
                unsigned short PD25IOR:1;         /*    PD25IOR   */
                unsigned short PD24IOR:1;         /*    PD24IOR   */
                unsigned short PD23IOR:1;         /*    PD23IOR   */
                unsigned short PD22IOR:1;         /*    PD22IOR   */
                unsigned short PD21IOR:1;         /*    PD21IOR   */
                unsigned short PD20IOR:1;         /*    PD20IOR   */
                unsigned short PD19IOR:1;         /*    PD19IOR   */
                unsigned short PD18IOR:1;         /*    PD18IOR   */
                unsigned short PD17IOR:1;         /*    PD17IOR   */
                unsigned short PD16IOR:1;         /*    PD16IOR   */
                } BIT;                            /*              */
        } PDIORH;                                 /*              */
    union {                                       /* PDIORL       */
        unsigned short WORD;                      /*  Word Access */
        struct {                                  /*  Bit Access  */
                unsigned short PD15IOR:1;         /*    PD15IOR   */
                unsigned short PD14IOR:1;         /*    PD14IOR   */
                unsigned short PD13IOR:1;         /*    PD13IOR   */
                unsigned short PD12IOR:1;         /*    PD12IOR   */
                unsigned short PD11IOR:1;         /*    PD11IOR   */
                unsigned short PD10IOR:1;         /*    PD10IOR   */
                unsigned short PD9IOR:1;          /*    PD9IOR    */
                unsigned short PD8IOR:1;          /*    PD8IOR    */
                unsigned short PD7IOR:1;          /*    PD7IOR    */
                unsigned short PD6IOR:1;          /*    PD6IOR    */
                unsigned short PD5IOR:1;          /*    PD5IOR    */
                unsigned short PD4IOR:1;          /*    PD4IOR    */
                unsigned short PD3IOR:1;          /*    PD3IOR    */
                unsigned short PD2IOR:1;          /*    PD2IOR    */
```

```
            unsigned short PD1IOR:1;          /*    PD1IOR    */
            unsigned short PD0IOR:1;          /*    PD0IOR    */
            } BIT;                            /*              */
      } PDIORL;                               /*              */
union {                                       /* PDCRH1       */
      unsigned short WORD;                    /*  Word Access */
      struct {                                /*  Bit Access  */
            unsigned short PD31MD:2;          /*    PD31MD    */
            unsigned short PD30MD:2;          /*    PD30MD    */
            unsigned short PD29MD:2;          /*    PD29MD    */
            unsigned short PD28MD:2;          /*    PD28MD    */
            unsigned short PD27MD:2;          /*    PD27MD    */
            unsigned short PD26MD:2;          /*    PD26MD    */
            unsigned short PD25MD:2;          /*    PD25MD    */
            unsigned short PD24MD:2;          /*    PD24MD    */
            } BIT;                            /*              */
      } PDCRH1;                               /*              */
union {                                       /* PDCRH2       */
      unsigned short WORD;                    /*  Word Access */
      struct {                                /*  Bit Access  */
            unsigned short PD23MD:2;          /*    PD23MD    */
            unsigned short PD22MD:2;          /*    PD22MD    */
            unsigned short PD21MD:2;          /*    PD21MD    */
            unsigned short PD20MD:2;          /*    PD20MD    */
            unsigned short PD19MD:2;          /*    PD19MD    */
            unsigned short PD18MD:2;          /*    PD18MD    */
            unsigned short PD17MD:2;          /*    PD17MD    */
            unsigned short PD16MD:2;          /*    PD16MD    */
            } BIT;                            /*              */
      } PDCRH2;                               /*              */
union {                                       /* PDCRL1       */
      unsigned short WORD;                    /*  Word Access */
      struct {                                /*  Bit Access  */
            unsigned short PD15MD0:1;         /*    PD15MD0   */
            unsigned short PD14MD0:1;         /*    PD14MD0   */
            unsigned short PD13MD0:1;         /*    PD13MD0   */
            unsigned short PD12MD0:1;         /*    PD12MD0   */
            unsigned short PD11MD0:1;         /*    PD11MD0   */
            unsigned short PD10MD0:1;         /*    PD10MD0   */
            unsigned short PD9MD0:1;          /*    PD9MD0    */
            unsigned short PD8MD0:1;          /*    PD8MD0    */
            unsigned short PD7MD0:1;          /*    PD7MD0    */
            unsigned short PD6MD0:1;          /*    PD6MD0    */
            unsigned short PD5MD0:1;          /*    PD5MD0    */
            unsigned short PD4MD0:1;          /*    PD4MD0    */
            unsigned short PD3MD0:1;          /*    PD3MD0    */
            unsigned short PD2MD0:1;          /*    PD2MD0    */
            unsigned short PD1MD0:1;          /*    PD1MD0    */
            unsigned short PD0MD0:1;          /*    PD0MD0    */
            } BIT;                            /*              */
      } PDCRL1;                               /*              */
```

```
        union {                                 /* PDCRL2       */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                    unsigned short PD15MD1:1;   /*    PD15MD1   */
                    unsigned short PD14MD1:1;   /*    PD14MD1   */
                    unsigned short PD13MD1:1;   /*    PD13MD1   */
                    unsigned short PD12MD1:1;   /*    PD12MD1   */
                    unsigned short PD11MD1:1;   /*    PD11MD1   */
                    unsigned short PD10MD1:1;   /*    PD10MD1   */
                    unsigned short PD9MD1:1;    /*    PD9MD1    */
                    unsigned short PD8MD1:1;    /*    PD8MD1    */
                    unsigned short PD7MD1:1;    /*    PD7MD1    */
                    unsigned short PD6MD1:1;    /*    PD6MD1    */
                    unsigned short PD5MD1:1;    /*    PD5MD1    */
                    unsigned short PD4MD1:1;    /*    PD4MD1    */
                    unsigned short PD3MD1:1;    /*    PD3MD1    */
                    unsigned short PD2MD1:1;    /*    PD2MD1    */
                    unsigned short PD1MD1:1;    /*    PD1MD1    */
                    unsigned short PD0MD1:1;    /*    PD0MD1    */
                    } BIT;                      /*              */
              } PDCRL2;                         /*              */
  };                                            /*              */
  struct st_porte {                             /* struct PORTE */
        union {                                 /* PEDRL        */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                    unsigned short PE15DR:1;    /*    PE15DR    */
                    unsigned short PE14DR:1;    /*    PE14DR    */
                    unsigned short PE13DR:1;    /*    PE13DR    */
                    unsigned short PE12DR:1;    /*    PE12DR    */
                    unsigned short PE11DR:1;    /*    PE11DR    */
                    unsigned short PE10DR:1;    /*    PE10DR    */
                    unsigned short PE9DR:1;     /*    PE9DR     */
                    unsigned short PE8DR:1;     /*    PE8DR     */
                    unsigned short PE7DR:1;     /*    PE7DR     */
                    unsigned short PE6DR:1;     /*    PE6DR     */
                    unsigned short PE5DR:1;     /*    PE5DR     */
                    unsigned short PE4DR:1;     /*    PE4DR     */
                    unsigned short PE3DR:1;     /*    PE3DR     */
                    unsigned short PE2DR:1;     /*    PE2DR     */
                    unsigned short PE1DR:1;     /*    PE1DR     */
                    unsigned short PE0DR:1;     /*    PE0DR     */
                    } BIT;                      /*              */
              } PEDRL;                          /*              */
        unsigned char wk0[2];                   /*              */
        union {                                 /* PEIORL       */
              unsigned short WORD;              /*  Word Access */
              struct {                          /*  Bit Access  */
                    unsigned short PE15IOR:1;   /*    PE15IOR   */
                    unsigned short PE14IOR:1;   /*    PE14IOR   */
                    unsigned short PE13IOR:1;   /*    PE13IOR   */
```

```
                unsigned short PE12IOR:1;          /*    PE12IOR   */
                unsigned short PE11IOR:1;          /*    PE11IOR   */
                unsigned short PE10IOR:1;          /*    PE10IOR   */
                unsigned short PE9IOR:1;           /*    PE9IOR    */
                unsigned short PE8IOR:1;           /*    PE8IOR    */
                unsigned short PE7IOR:1;           /*    PE7IOR    */
                unsigned short PE6IOR:1;           /*    PE6IOR    */
                unsigned short PE5IOR:1;           /*    PE5IOR    */
                unsigned short PE4IOR:1;           /*    PE4IOR    */
                unsigned short PE3IOR:1;           /*    PE3IOR    */
                unsigned short PE2IOR:1;           /*    PE2IOR    */
                unsigned short PE1IOR:1;           /*    PE1IOR    */
                unsigned short PE0IOR:1;           /*    PE0IOR    */
                } BIT;                             /*              */
          } PEIORL;                                /*              */
      unsigned char wk1[2];                        /*              */
      union {                                      /* PECRL1       */
          unsigned short WORD;                     /*  Word Access */
          struct {                                 /*  Bit Access  */
                unsigned short PE15MD:2;           /*    PE15MD    */
                unsigned short PE14MD:2;           /*    PE14MD    */
                unsigned short PE13MD:2;           /*    PE13MD    */
                unsigned short PE12MD:2;           /*    PE12MD    */
                unsigned short PE11MD:2;           /*    PE11MD    */
                unsigned short PE10MD:2;           /*    PE10MD    */
                unsigned short PE9MD:2;            /*    PE9MD     */
                unsigned short PE8MD:2;            /*    PE8MD     */
                } BIT;                             /*              */
          } PECRL1;                                /*              */
      union {                                      /* PECRL2       */
          unsigned short WORD;                     /*  Word Access */
          struct {                                 /*  Bit Access  */
                unsigned short PE7MD:2;            /*    PE7MD     */
                unsigned short PE6MD:2;            /*    PE6MD     */
                unsigned short PE5MD:2;            /*    PE5MD     */
                unsigned short PE4MD:2;            /*    PE4MD     */
                unsigned short PE3MD:2;            /*    PE3MD     */
                unsigned short PE2MD:2;            /*    PE2MD     */
                unsigned short PE1MD:2;            /*    PE1MD     */
                unsigned short PE0MD:2;            /*    PE0MD     */
                } BIT;                             /*              */
          } PECRL2;                                /*              */
};                                                 /*              */
struct st_portf {                                  /* struct PORTF */
      union {                                      /* PFDR         */
          unsigned short WORD;                     /*  Word Access */
          struct {                                 /*  Bit Access  */
                unsigned short :8;                 /*              */
                unsigned short PF7DR:1;            /*    PF7DR     */
                unsigned short PF6DR:1;            /*    PF6DR     */
                unsigned short PF5DR:1;            /*    PF5DR     */
```

```
                    unsigned short PF4DR:1;          /*    PF4DR     */
                    unsigned short PF3DR:1;          /*    PF3DR     */
                    unsigned short PF2DR:1;          /*    PF2DR     */
                    unsigned short PF1DR:1;          /*    PF1DR     */
                    unsigned short PF0DR:1;          /*    PF0DR     */
                    } BIT;                           /*              */
             } PFDR;                                 /*              */
};                                                   /*              */
struct st_mtu {                                      /* struct MTU   */
       union {                                       /* ICSR1        */
             unsigned short WORD;                    /*  Word Access */
             struct {                                /*  Bit Access  */
                    unsigned short POE3F:1;          /*    POE3F     */
                    unsigned short POE2F:1;          /*    POE2F     */
                    unsigned short POE1F:1;          /*    POE1F     */
                    unsigned short POE0F:1;          /*    POE0F     */
                    unsigned short :3;               /*              */
                    unsigned short PIE:1;            /*    PIE       */
                    unsigned short POE3M:2;          /*    POE3M     */
                    unsigned short POE2M:2;          /*    POE2M     */
                    unsigned short POE1M:2;          /*    POE1M     */
                    unsigned short POE0M:2;          /*    POE0M     */
                    } BIT;                           /*              */
             } ICSR1;                                /*              */
       union {                                       /* OCSR         */
             unsigned short WORD;                    /*  Word Access */
             struct {                                /*  Bit Access  */
                    unsigned short OSF:1;            /*    OSF       */
                    unsigned short :5;               /*              */
                    unsigned short OCE:1;            /*    OCE       */
                    unsigned short OIE:1;            /*    OIE       */
                    unsigned short :8;               /*              */
                    } BIT;                           /*              */
             } OCSR;                                 /*              */
};                                                   /*              */
struct st_cmt {                                      /* struct CMT   */
       union {                                       /* CMSTR        */
             unsigned short WORD;                    /*  Word Access */
             struct {                                /*  Bit Access  */
                    unsigned short :14;              /*              */
                    unsigned short STR:2;            /*    STR       */
                    } BIT;                           /*              */
             } CMSTR;                                /*              */
       union {                                       /* CMCSR_0      */
             unsigned short WORD;                    /*  Word Access */
             struct {                                /*  Bit Access  */
                    unsigned short :8;               /*              */
                    unsigned short CMF:1;            /*    CMF       */
                    unsigned short CMIE:1;           /*    CMIE      */
                    unsigned short :4;               /*              */
                    unsigned short CKS:2;            /*    CKS       */
```

```
                } BIT;                                  /*              */
            } CMCSR_0;                                  /*              */
        unsigned short CMCNT_0;                         /* CMCNT_0      */
        unsigned short CMCOR_0;                         /* CMCOR_0      */
        union {                                         /* CMCSR_1      */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short :8;                      /*              */
                unsigned short CMF:1;                   /*    CMF       */
                unsigned short CMIE:1;                  /*    CMIE      */
                unsigned short :4;                      /*              */
                unsigned short CKS:2;                   /*    CKS       */
                } BIT;                                  /*              */
            } CMCSR_1;                                  /*              */
        unsigned short CMCNT_1;                         /* CMCNT_1      */
        unsigned short CMCOR_1;                         /* CMCOR_1      */
};                                                      /*              */
struct st_ad {                                          /* struct A/D   */
        union {                                         /* ADDR0        */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short AD:10;                   /*    AD        */
                unsigned short :6;                      /*              */
                } BIT;                                  /*              */
            } ADDR0;                                    /*              */
        union {                                         /* ADDR1        */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short AD:10;                   /*    AD        */
                unsigned short :6;                      /*              */
                } BIT;                                  /*              */
            } ADDR1;                                    /*              */
        union {                                         /* ADDR2        */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short AD:10;                   /*    AD        */
                unsigned short :6;                      /*              */
                } BIT;                                  /*              */
            } ADDR2;                                    /*              */
        union {                                         /* ADDR3        */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short AD:10;                   /*    AD        */
                unsigned short :6;                      /*              */
                } BIT;                                  /*              */
            } ADDR3;                                    /*              */
        union {                                         /* ADDR4        */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short AD:10;                   /*    AD        */
                unsigned short :6;                      /*              */
```

```
            } BIT;                              /*              */
      } ADDR4;                                  /*              */
union {                                         /* ADDR5        */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
            unsigned short AD:10;               /*    AD        */
            unsigned short :6;                  /*              */
            } BIT;                              /*              */
      } ADDR5;                                  /*              */
union {                                         /* ADDR6        */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
            unsigned short AD:10;               /*    AD        */
            unsigned short :6;                  /*              */
            } BIT;                              /*              */
      } ADDR6;                                  /*              */
union {                                         /* ADDR7        */
      unsigned short WORD;                      /*  Word Access */
      struct {                                  /*  Bit Access  */
            unsigned short AD:10;               /*    AD        */
            unsigned short :6;                  /*              */
            } BIT;                              /*              */
      } ADDR7;                                  /*              */
unsigned char wk0[80];                          /*              */
union {                                         /* ADCSR_0      */
      unsigned char BYTE;                       /*  Byte Access */
      struct {                                  /*  Bit Access  */
            unsigned char ADF:1;                /*    ADF       */
            unsigned char ADIE:1;               /*    ADIE      */
            unsigned char :1;                   /*              */
            unsigned char ADM:1;                /*    ADM       */
            unsigned char :2;                   /*              */
            unsigned char CH:2;                 /*    CH        */
            } BIT;                              /*              */
      } ADCSR_0;                                /*              */
union {                                         /* ADCSR_1      */
      unsigned char BYTE;                       /*  Byte Access */
      struct {                                  /*  Bit Access  */
            unsigned char ADF:1;                /*    ADF       */
            unsigned char ADIE:1;               /*    ADIE      */
            unsigned char :1;                   /*              */
            unsigned char ADM:1;                /*    ADM       */
            unsigned char :2;                   /*              */
            unsigned char CH:2;                 /*    CH        */
            } BIT;                              /*              */
      } ADCSR_1;                                /*              */
unsigned char wk1[6];                           /*              */
union {                                         /* ADCR_0       */
      unsigned char BYTE;                       /*  Byte Access */
      struct {                                  /*  Bit Access  */
            unsigned char TRGE:1;               /*    TRGE      */
```

```
                unsigned char CKS:2;            /*    CKS       */
                unsigned char ADST:1;           /*    ADST      */
                unsigned char ADCS:1;           /*    ADCS      */
                unsigned char :3;               /*              */
                } BIT;                          /*              */
          } ADCR_0;                             /*              */
      union {                                   /* ADCR_1       */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char TRGE:1;           /*    TRGE      */
                unsigned char CKS:2;            /*    CKS       */
                unsigned char ADST:1;           /*    ADST      */
                unsigned char ADCS:1;           /*    ADCS      */
                unsigned char :3;               /*              */
                } BIT;                          /*              */
          } ADCR_1;                             /*              */
      unsigned char wk2[874];                   /*              */
      union {                                   /* ADTSR        */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char :4;               /*              */
                unsigned char TRG1S:2;          /*    TRG1S     */
                unsigned char TRG0S:2;          /*    TRG0S     */
                } BIT;                          /*              */
          } ADTSR;                              /*              */
};                                              /*              */
struct st_flash {                               /* struct FLASH */
      union {                                   /* FLMCR1       */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char FWE:1;            /*    FWE       */
                unsigned char SWE:1;            /*    SWE       */
                unsigned char ESU:1;            /*    ESU       */
                unsigned char PSU:1;            /*    PSU       */
                unsigned char EV:1;             /*    EV        */
                unsigned char PV:1;             /*    PV        */
                unsigned char E:1;              /*    E         */
                unsigned char P:1;              /*    P         */
                } BIT;                          /*              */
          } FLMCR1;                             /*              */
      union {                                   /* FLMCR2       */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char FLER:1;           /*    FLER      */
                unsigned char :7;               /*              */
                } BIT;                          /*              */
          } FLMCR2;                             /*              */
      union {                                   /* EBR1         */
          unsigned char BYTE;                   /*  Byte Access */
          struct {                              /*  Bit Access  */
                unsigned char EB:8;             /*    EB        */
```

```
                } BIT;                                  /*              */
            } EBR1;                                     /*              */
        union {                                         /* EBR2         */
            unsigned char BYTE;                         /*  Byte Access */
            struct {                                    /*  Bit Access  */
                unsigned char :4;                       /*              */
                unsigned char EB11:1;                   /*    EB11      */
                unsigned char EB10:1;                   /*    EB10      */
                unsigned char EB9:1;                    /*    EB9       */
                unsigned char EB8:1;                    /*    EB8       */
                } BIT;                                  /*              */
            } EBR2;                                     /*              */
        unsigned char wk0[164];                         /*              */
        union {                                         /* RAMER        */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short :12;                     /*              */
                unsigned short RAMS:1;                  /*    RAMS      */
                unsigned short RAM:3;                   /*    RAM       */
                } BIT;                                  /*              */
            } RAMER;                                    /*              */
};                                                      /*              */
struct st_ubc {                                         /* struct UBC   */
        union {                                         /* UBARH        */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short UBA31:1;                 /*    UBA31     */
                unsigned short UBA30:1;                 /*    UBA30     */
                unsigned short UBA29:1;                 /*    UBA29     */
                unsigned short UBA28:1;                 /*    UBA28     */
                unsigned short UBA27:1;                 /*    UBA27     */
                unsigned short UBA26:1;                 /*    UBA26     */
                unsigned short UBA25:1;                 /*    UBA25     */
                unsigned short UBA24:1;                 /*    UBA24     */
                unsigned short UBA23:1;                 /*    UBA23     */
                unsigned short UBA22:1;                 /*    UBA22     */
                unsigned short UBA21:1;                 /*    UBA21     */
                unsigned short UBA20:1;                 /*    UBA20     */
                unsigned short UBA19:1;                 /*    UBA19     */
                unsigned short UBA18:1;                 /*    UBA18     */
                unsigned short UBA17:1;                 /*    UBA17     */
                unsigned short UBA16:1;                 /*    UBA16     */
                } BIT;                                  /*              */
            } UBARH;                                    /*              */
        union {                                         /* UBARL        */
            unsigned short WORD;                        /*  Word Access */
            struct {                                    /*  Bit Access  */
                unsigned short UBA15:1;                 /*    UBA15     */
                unsigned short UBA14:1;                 /*    UBA14     */
                unsigned short UBA13:1;                 /*    UBA13     */
                unsigned short UBA12:1;                 /*    UBA12     */
```

```
            unsigned short UBA11:1;          /*    UBA11     */
            unsigned short UBA10:1;          /*    UBA10     */
            unsigned short UBA9:1;           /*    UBA9      */
            unsigned short UBA8:1;           /*    UBA8      */
            unsigned short UBA7:1;           /*    UBA7      */
            unsigned short UBA6:1;           /*    UBA6      */
            unsigned short UBA5:1;           /*    UBA5      */
            unsigned short UBA4:1;           /*    UBA4      */
            unsigned short UBA3:1;           /*    UBA3      */
            unsigned short UBA2:1;           /*    UBA2      */
            unsigned short UBA1:1;           /*    UBA1      */
            unsigned short UBA0:1;           /*    UBA0      */
            } BIT;                           /*              */
      } UBARL;                               /*              */
union {                                      /* UBAMRH       */
      unsigned short WORD;                   /*  Word Access */
      struct {                               /*  Bit Access  */
            unsigned short UBM31:1;          /*    UBM31     */
            unsigned short UBM30:1;          /*    UBM30     */
            unsigned short UBM29:1;          /*    UBM29     */
            unsigned short UBM28:1;          /*    UBM28     */
            unsigned short UBM27:1;          /*    UBM27     */
            unsigned short UBM26:1;          /*    UBM26     */
            unsigned short UBM25:1;          /*    UBM25     */
            unsigned short UBM24:1;          /*    UBM24     */
            unsigned short UBM23:1;          /*    UBM23     */
            unsigned short UBM22:1;          /*    UBM22     */
            unsigned short UBM21:1;          /*    UBM21     */
            unsigned short UBM20:1;          /*    UBM20     */
            unsigned short UBM19:1;          /*    UBM19     */
            unsigned short UBM18:1;          /*    UBM18     */
            unsigned short UBM17:1;          /*    UBM17     */
            unsigned short UBM16:1;          /*    UBM16     */
            } BIT;                           /*              */
      } UBAMRH;                              /*              */
union {                                      /* UBAMRL       */
      unsigned short WORD;                   /*  Word Access */
      struct {                               /*  Bit Access  */
            unsigned short UBM15:1;          /*    UBM15     */
            unsigned short UBM14:1;          /*    UBM14     */
            unsigned short UBM13:1;          /*    UBM13     */
            unsigned short UBM12:1;          /*    UBM12     */
            unsigned short UBM11:1;          /*    UBM11     */
            unsigned short UBM10:1;          /*    UBM10     */
            unsigned short UBM9:1;           /*    UBM9      */
            unsigned short UBM8:1;           /*    UBM8      */
            unsigned short UBM7:1;           /*    UBM7      */
            unsigned short UBM6:1;           /*    UBM6      */
            unsigned short UBM5:1;           /*    UBM5      */
            unsigned short UBM4:1;           /*    UBM4      */
            unsigned short UBM3:1;           /*    UBM3      */
```

```
                    unsigned short UBM2:1;          /*    UBM2      */
                    unsigned short UBM1:1;          /*    UBM1      */
                    unsigned short UBM0:1;          /*    UBM0      */
                    } BIT;                          /*              */
              } UBAMRL;                             /*              */
        union {                                     /* UBBR         */
              unsigned short WORD;                  /*  Word Access */
              struct {                              /*  Bit Access  */
                    unsigned short :8;              /*              */
                    unsigned short CP:2;            /*    CP        */
                    unsigned short ID:2;            /*    ID        */
                    unsigned short RW:2;            /*    RW        */
                    unsigned short SZ:2;            /*    SZ        */
                    } BIT;                          /*              */
              } UBBR;                               /*              */
        union {                                     /* UBCR         */
              unsigned short WORD;                  /*  Word Access */
              struct {                              /*  Bit Access  */
                    unsigned short :15;             /*              */
                    unsigned short UBID:1;          /*    UBID      */
                    } BIT;                          /*              */
              } UBCR;                               /*              */
  };                                                /*              */
  struct st_wdt {                                   /* struct WDT   */
        union {                                     /* TCSR         */
              unsigned char BYTE;                   /*  Byte Access */
              struct {                              /*  Bit Access  */
                    unsigned char OVF:1;            /*    OVF       */
                    unsigned char WTIT:1;           /*    WT/IT     */
                    unsigned char TME:1;            /*    TME       */
                    unsigned char :2;               /*              */
                    unsigned char CKS:3;            /*    CKS       */
                    } BIT;                          /*              */
              } TCSR;                               /*              */
        unsigned char TCNT;                         /* TCNT         */
        union {                                     /* RSTCSR       */
              unsigned char BYTE;                   /*  Byte Access */
              struct {                              /*  Bit Access  */
                    unsigned char WOVF:1;           /*    WOVF      */
                    unsigned char RSTE:1;           /*    RSTE      */
                    unsigned char RSTS:1;           /*    RSTS      */
                    unsigned char :5;               /*              */
                    } BIT;                          /*              */
              } RSTCSR;                             /*              */
  };                                                /*              */
  struct st_stby {                                  /* struct STBY  */
        union {                                     /* SBYCR        */
              unsigned char BYTE;                   /*  Byte Access */
              struct {                              /*  Bit Access  */
                    unsigned char SBY:1;            /*    SBY       */
                    unsigned char HIZ:1;            /*    HIZ       */
```

```
                unsigned char :4;                   /*              */
                unsigned char IRQEH:1;              /*    IRQEH     */
                unsigned char IRQEL:1;              /*    IRQEL     */
                } BIT;                              /*              */
          } SBYCR;                                  /*              */
    unsigned char wk0[3];                           /*              */
    union {                                         /* SYSCR        */
          unsigned char BYTE;                       /*  Byte Access */
          struct {                                  /*  Bit Access  */
                unsigned char :6;                   /*              */
                unsigned char AUDSRST:1;            /*    AUDSRST   */
                unsigned char RAME:1;               /*    RAME      */
                } BIT;                              /*              */
          } SYSCR;                                  /*              */
    unsigned char wk1[3];                           /*              */
    union {                                         /* MSTCR1       */
          unsigned short WORD;                      /*  Word Access */
          struct {                                  /*  Bit Access  */
                unsigned short :4;                  /*              */
                unsigned short MSTP27:1;            /*    MSTP27    */
                unsigned short MSTP26:1;            /*    MSTP26    */
                unsigned short MSTP25:1;            /*    MSTP25    */
                unsigned short MSTP24:1;            /*    MSTP24    */
                unsigned short :2;                  /*              */
                unsigned short MSTP21:1;            /*    MSTP21    */
                unsigned short :1;                  /*              */
                unsigned short MSTP19:1;            /*    MSTP19    */
                unsigned short MSTP18:1;            /*    MSTP18    */
                unsigned short MSTP17:1;            /*    MSTP17    */
                unsigned short MSTP16:1;            /*    MSTP16    */
                } BIT;                              /*              */
          } MSTCR1;                                 /*              */
    union {                                         /* MSTCR2       */
          unsigned short WORD;                      /*  Word Access */
          struct {                                  /*  Bit Access  */
                unsigned short :2;                  /*              */
                unsigned short MSTP13:1;            /*    MSTP13    */
                unsigned short MSTP12:1;            /*    MSTP12    */
                unsigned short :6;                  /*              */
                unsigned short MSTP5:1;             /*    MSTP5     */
                unsigned short MSTP4:1;             /*    MSTP4     */
                unsigned short MSTP3:1;             /*    MSTP3     */
                unsigned short MSTP2:1;             /*    MSTP2     */
                unsigned short :1;                  /*              */
                unsigned short MSTP0:1;             /*    MSTP0     */
                } BIT;                              /*              */
          } MSTCR2;                                 /*              */
};                                                  /*              */
struct st_bsc {                                     /* struct BSC   */
    union {                                         /* BCR1         */
          unsigned short WORD;                      /*  Word Access */
```

```
            struct {                                /*  Bit Access  */
                   unsigned short :2;               /*              */
                   unsigned short MTURWE:1;         /*    MTURWE    */
                   unsigned short :5;               /*              */
                   unsigned short A3LG:1;           /*    A3LG      */
                   unsigned short A2LG:1;           /*    A2LG      */
                   unsigned short A1LG:1;           /*    A1LG      */
                   unsigned short A0LG:1;           /*    A0LG      */
                   unsigned short A3SZ:1;           /*    A3SZ      */
                   unsigned short A2SZ:1;           /*    A2SZ      */
                   unsigned short A1SZ:1;           /*    A1SZ      */
                   unsigned short A0SZ:1;           /*    A0SZ      */
                   } BIT;                           /*              */
            } BCR1;                                 /*              */
      union {                                       /* BCR2         */
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                   unsigned short IW3:2;            /*    IW3       */
                   unsigned short IW2:2;            /*    IW2       */
                   unsigned short IW1:2;            /*    IW1       */
                   unsigned short IW0:2;            /*    IW0       */
                   unsigned short CW3:1;            /*    CW3       */
                   unsigned short CW2:1;            /*    CW2       */
                   unsigned short CW1:1;            /*    CW1       */
                   unsigned short CW0:1;            /*    CW0       */
                   unsigned short SW3:1;            /*    SW3       */
                   unsigned short SW2:1;            /*    SW2       */
                   unsigned short SW1:1;            /*    SW1       */
                   unsigned short SW0:1;            /*    SW0       */
                   } BIT;                           /*              */
            } BCR2;                                 /*              */
      union {                                       /* WCR1         */
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                   unsigned short W3:4;             /*    W3        */
                   unsigned short W2:4;             /*    W2        */
                   unsigned short W1:4;             /*    W1        */
                   unsigned short W0:4;             /*    W0        */
                   } BIT;                           /*              */
            } WCR1;                                 /*              */
      union {                                       /* WCR2         */
            unsigned short WORD;                    /*  Word Access */
            struct {                                /*  Bit Access  */
                   unsigned short :12;              /*              */
                   unsigned short DSW:4;            /*    DSW       */
                   } BIT;                           /*              */
            } WCR2;                                 /*              */
};                                                  /*              */
struct st_dmac {                                    /* struct DMAC  */
      union {                                       /* DMAOR        */
            unsigned short WORD;                    /*  Word Access */
```

```
            struct {                                /*  Bit Access  */
                  unsigned short :6;                /*              */
                  unsigned short PR:2;              /*    PR        */
                  unsigned short :5;                /*              */
                  unsigned short AE:1;              /*    AE        */
                  unsigned short NMIF:1;            /*    NMIF      */
                  unsigned short DME:1;             /*    DME       */
                  } BIT;                            /*              */
            } DMAOR;                                /*              */
};                                                  /*              */
struct st_dmac0 {                                   /* struct DMAC0 */
      unsigned long SAR0;                           /* SAR0         */
      unsigned long DAR0;                           /* DAR0         */
      unsigned long DMATCR0;                        /* DMATCR0      */
      union {                                       /* CHCR0        */
            unsigned long LONG;                     /*  Long Access */
            struct {                                /*  Bit Access  */
                  unsigned long :13;                /*              */
                  unsigned long RL:1;               /*    RL        */
                  unsigned long AM:1;               /*    AM        */
                  unsigned long AL:1;               /*    AL        */
                  unsigned long DM:2;               /*    DM        */
                  unsigned long SM:2;               /*    SM        */
                  unsigned long RS:4;               /*    RS        */
                  unsigned long :1;                 /*              */
                  unsigned long DS:1;               /*    DS        */
                  unsigned long TM:1;               /*    TM        */
                  unsigned long TS:2;               /*    TS        */
                  unsigned long IE:1;               /*    IE        */
                  unsigned long TE:1;               /*    TE        */
                  unsigned long DE:1;               /*    DE        */
                  } BIT;                            /*              */
            } CHCR0;                                /*              */
};                                                  /*              */
struct st_dmac1 {                                   /* struct DMAC1 */
      unsigned long SAR1;                           /* SAR1         */
      unsigned long DAR1;                           /* DAR1         */
      unsigned long DMATCR1;                        /* DMATCR1      */
      union {                                       /* CHCR1        */
            unsigned long LONG;                     /*  Long Access */
            struct {                                /*  Bit Access  */
                  unsigned long :13;                /*              */
                  unsigned long RL:1;               /*    RL        */
                  unsigned long AM:1;               /*    AM        */
                  unsigned long AL:1;               /*    AL        */
                  unsigned long DM:2;               /*    DM        */
                  unsigned long SM:2;               /*    SM        */
                  unsigned long RS:4;               /*    RS        */
                  unsigned long :1;                 /*              */
                  unsigned long DS:1;               /*    DS        */
                  unsigned long TM:1;               /*    TM        */
```

```
                    unsigned long TS:2;             /*    TS        */
                    unsigned long IE:1;             /*    IE        */
                    unsigned long TE:1;             /*    TE        */
                    unsigned long DE:1;             /*    DE        */
                    } BIT;                          /*              */
              } CHCR1;                              /*              */
};                                                  /*              */
struct st_dmac2 {                                   /* struct DMAC2 */
       unsigned long SAR2;                          /* SAR2         */
       unsigned long DAR2;                          /* DAR2         */
       unsigned long DMATCR2;                       /* DMATCR2      */
       union {                                      /* CHCR2        */
             unsigned long LONG;                    /*  Long Access */
             struct {                               /*  Bit Access  */
                    unsigned long :12;              /*              */
                    unsigned long RO:1;             /*    RO        */
                    unsigned long :3;               /*              */
                    unsigned long DM:2;             /*    DM        */
                    unsigned long SM:2;             /*    SM        */
                    unsigned long RS:4;             /*    RS        */
                    unsigned long :2;               /*              */
                    unsigned long TM:1;             /*    TM        */
                    unsigned long TS:2;             /*    TS        */
                    unsigned long IE:1;             /*    IE        */
                    unsigned long TE:1;             /*    TE        */
                    unsigned long DE:1;             /*    DE        */
                    } BIT;                          /*              */
             } CHCR2;                               /*              */
};                                                  /*              */
struct st_dmac3 {                                   /* struct DMAC3 */
       unsigned long SAR3;                          /* SAR3         */
       unsigned long DAR3;                          /* DAR3         */
       unsigned long DMATCR3;                       /* DMATCR3      */
       union {                                      /* CHCR3        */
             unsigned long LONG;                    /*  Long Access */
             struct {                               /*  Bit Access  */
                    unsigned long :11;              /*              */
                    unsigned long DI:1;             /*    DI        */
                    unsigned long :4;               /*              */
                    unsigned long DM:2;             /*    DM        */
                    unsigned long SM:2;             /*    SM        */
                    unsigned long RS:4;             /*    RS        */
                    unsigned long :2;               /*              */
                    unsigned long TM:1;             /*    TM        */
                    unsigned long TS:2;             /*    TS        */
                    unsigned long IE:1;             /*    IE        */
                    unsigned long TE:1;             /*    TE        */
                    unsigned long DE:1;             /*    DE        */
                    } BIT;                          /*              */
             } CHCR3;                               /*              */
};                                                  /*              */
```

```
struct st_dtc {                                 /* struct DTC   */
       union {                                  /* DTEA         */
             unsigned char BYTE;                /*  Byte Access */
             struct {                           /*  Bit Access  */
                    unsigned char DTEA:8;       /*    DTEA      */
                    } BIT;                      /*              */
             } DTEA;                            /*              */
       union {                                  /* DTEB         */
             unsigned char BYTE;                /*  Byte Access */
             struct {                           /*  Bit Access  */
                    unsigned char DTEB:8;       /*    DTEB      */
                    } BIT;                      /*              */
             } DTEB;                            /*              */
       union {                                  /* DTEC         */
             unsigned char BYTE;                /*  Byte Access */
             struct {                           /*  Bit Access  */
                    unsigned char DTEC:8;       /*    DTEC      */
                    } BIT;                      /*              */
             } DTEC;                            /*              */
       union {                                  /* DTED         */
             unsigned char BYTE;                /*  Byte Access */
             struct {                           /*  Bit Access  */
                    unsigned char DTED:8;       /*    DTED      */
                    } BIT;                      /*              */
             } DTED;                            /*              */
       unsigned char wk0[2];                    /*              */
       union {                                  /* DTCSR        */
             unsigned short WORD;               /*  Word Access */
             struct {                           /*  Bit Access  */
                    unsigned short :5;          /*              */
                    unsigned short NMIF:1;      /*    NMIF      */
                    unsigned short AE:1;        /*    AE        */
                    unsigned short SWDTE:1;     /*    SWDTE     */
                    unsigned short DTVEC7:1;    /*    DTVEC7    */
                    unsigned short DTVEC6:1;    /*    DTVEC6    */
                    unsigned short DTVEC5:1;    /*    DTVEC5    */
                    unsigned short DTVEC4:1;    /*    DTVEC4    */
                    unsigned short DTVEC3:1;    /*    DTVEC3    */
                    unsigned short DTVEC2:1;    /*    DTVEC2    */
                    unsigned short DTVEC1:1;    /*    DTVEC1    */
                    unsigned short DTVEC0:1;    /*    DTVEC0    */
                    } BIT;                      /*              */
             } DTCSR;                           /*              */
       unsigned short DTBR;                     /* DTBR         */
       unsigned char wk1[6];                    /*              */
       union {                                  /* DTEE         */
             unsigned char BYTE;                /*  Byte Access */
             struct {                           /*  Bit Access  */
                    unsigned char :2;           /*              */
                    unsigned char DTEE5:1;      /*    DTEE5     */
                    unsigned char :1;           /*              */
```

```
                    unsigned char DTEE3:1;          /*    DTEE3     */
                    unsigned char DTEE2:1;          /*    DTEE2     */
                    unsigned char DTEE1:1;          /*    DTEE1     */
                    unsigned char DTEE0:1;          /*    DTEE0     */
                    } BIT;                          /*              */
              } DTEE;                               /*              */
        unsigned char wk2[1];                       /*              */
        union {                                     /* DTEG         */
              unsigned char BYTE;                   /*  Byte Access */
              struct {                              /*  Bit Access  */
                    unsigned char DTEG7:1;          /*    DTEG7     */
                    unsigned char :7;               /*              */
                    } BIT;                          /*              */
              } DTEG;                               /*              */
  };                                                /*              */
  struct st_iic {                                   /* struct IIC   */
        union {                                     /* SCRX         */
              unsigned char BYTE;                   /*  Byte Access */
              struct {                              /*  Bit Access  */
                    unsigned char :2;               /*              */
                    unsigned char IICX0:1;          /*    IICX0     */
                    unsigned char IICE:1;           /*    IICE      */
                    unsigned char HNDS:1;           /*    HNDS      */
                    unsigned char :1;               /*              */
                    unsigned char ICDRF0:1;         /*    ICDRF0    */
                    unsigned char STOPIM:1;         /*    STOPIM    */
                    } BIT;                          /*              */
              } SCRX;                               /*              */
        unsigned char wk0[23];                      /*              */
        union {                                     /* ICCR0        */
              unsigned char BYTE;                   /*  Byte Access */
              struct {                              /*  Bit Access  */
                    unsigned char ICE:1;            /*    ICE       */
                    unsigned char IEIC:1;           /*    IEIC      */
                    unsigned char MST:1;            /*    MST       */
                    unsigned char TRS:1;            /*    TRS       */
                    unsigned char ACKE:1;           /*    ACKE      */
                    unsigned char BBSY:1;           /*    BBSY      */
                    unsigned char IRIC:1;           /*    IRIC      */
                    unsigned char SCP:1;            /*    SCP       */
                    } BIT;                          /*              */
              } ICCR0;                              /*              */
        union {                                     /* ICSR0        */
              unsigned char BYTE;                   /*  Byte Access */
              struct {                              /*  Bit Access  */
                    unsigned char ESTP:1;           /*    ESTP      */
                    unsigned char STOP:1;           /*    STOP      */
                    unsigned char IRTR:1;           /*    IRTR      */
                    unsigned char AASX:1;           /*    AASX      */
                    unsigned char AL:1;             /*    AL        */
                    unsigned char AAS:1;            /*    AAS       */
```

```
                    unsigned char ADZ:1;              /*    ADZ       */
                    unsigned char ACKB:1;             /*    ACKB      */
                    } BIT;                            /*              */
              } ICSR0;                                /*              */
        unsigned char wk1[4];                         /*              */
        union {                                       /* ICDR0        */
              unsigned char BYTE;                     /*  Byte Access */
              struct {                                /*  Bit Access  */
                    unsigned char ICDR:8;             /*    ICDR      */
                    } BIT;                            /*              */
              } ICDR0;                                /*              */
        union {                                       /* ICMR0        */
              unsigned char BYTE;                     /*  Byte Access */
              struct {                                /*  Bit Access  */
                    unsigned char MLS:1;              /*    MLS       */
                    unsigned char WAIT:1;             /*    WAIT      */
                    unsigned char CKS:3;              /*    CKS       */
                    unsigned char BC:3;               /*    BC        */
                    } BIT;                            /*              */
              } ICMR0;                                /*              */
};                                                    /*              */
struct st_hudi {                                      /* struct H-UDI */
        union {                                       /* SDIR         */
              unsigned short WORD;                    /*  Word Access */
              struct {                                /*  Bit Access  */
                    unsigned short TS:4;              /*    TS        */
                    unsigned short :12;               /*              */
                    } BIT;                            /*              */
              } SDIR;                                 /*              */
        union {                                       /* SDSR         */
              unsigned short WORD;                    /*  Word Access */
              struct {                                /*  Bit Access  */
                    unsigned short :15;               /*              */
                    unsigned short SDTRF:1;           /*    SDTRF     */
                    } BIT;                            /*              */
              } SDSR;                                 /*              */
        unsigned short SDDRH;                         /* SDDRH        */
        unsigned short SDDRL;                         /* SDDRL        */
};                                                    /*              */
#define P_SCI0  (*(volatile struct st_sci0 *)0xFFFF81A0) /* SCI0 Address  */
#define P_SCI1  (*(volatile struct st_sci1 *)0xFFFF81B0) /* SCI1 Address  */
#define P_SCI2  (*(volatile struct st_sci2 *)0xFFFF81C0) /* SCI2 Address  */
#define P_SCI3  (*(volatile struct st_sci3 *)0xFFFF81D0) /* SCI3 Address  */
#define P_MTU34 (*(volatile struct st_mtu34 *)0xFFFF8200)/* MTU34 Address */
#define P_MTU0  (*(volatile struct st_mtu0 *)0xFFFF8260) /* MTU0 Address  */
#define P_MTU1  (*(volatile struct st_mtu1 *)0xFFFF8280) /* MTU1 Address  */
#define P_MTU2  (*(volatile struct st_mtu2 *)0xFFFF82A0) /* MTU2 Address  */
#define P_INTC  (*(volatile struct st_intc *)0xFFFF8348) /* INTC Address  */
#define P_PORTA (*(volatile struct st_porta *)0xFFFF8380)/* PORTA Address */
#define P_PORTB (*(volatile struct st_portb *)0xFFFF8390)/* PORTB Address */
#define P_PORTC (*(volatile struct st_portc *)0xFFFF8392)/* PORTC Address */
```

```
#define P_PORTD (*(volatile struct st_portd *)0xFFFF83A0)/* PORTD Address */
#define P_PORTE (*(volatile struct st_porte *)0xFFFF83B0)/* PORTE Address */
#define P_PORTF (*(volatile struct st_portf *)0xFFFF83B2)/* PORTF Address */
#define P_MTU   (*(volatile struct st_mtu *)0xFFFF83C0)  /* MTU Address   */
#define P_CMT   (*(volatile struct st_cmt *)0xFFFF83D0)  /* CMT Address   */
#define P_AD    (*(volatile struct st_ad *)0xFFFF8420)   /* A/D Address   */
#define P_FLASH (*(volatile struct st_flash *)0xFFFF8580)/* FLASH Address */
#define P_UBC   (*(volatile struct st_ubc *)0xFFFF8600)  /* UBC Address   */
#define P_WDT   (*(volatile struct st_wdt *)0xFFFF8610)  /* WDT Address   */
#define P_STBY  (*(volatile struct st_stby *)0xFFFF8614) /* STBY Address  */
#define P_BSC   (*(volatile struct st_bsc *)0xFFFF8620)  /* BSC Address   */
#define P_DMAC  (*(volatile struct st_dmac *)0xFFFF86B0) /* DMAC Address  */
#define P_DMAC0 (*(volatile struct st_dmac0 *)0xFFFF86C0)/* DMAC0 Address */
#define P_DMAC1 (*(volatile struct st_dmac1 *)0xFFFF86D0)/* DMAC1 Address */
#define P_DMAC2 (*(volatile struct st_dmac2 *)0xFFFF86E0)/* DMAC2 Address */
#define P_DMAC3 (*(volatile struct st_dmac3 *)0xFFFF86F0)/* DMAC3 Address */
#define P_DTC   (*(volatile struct st_dtc *)0xFFFF8700)  /* DTC Address   */
#define P_IIC   (*(volatile struct st_iic *)0xFFFF87F0)  /* IIC Address   */
#define P_HUDI  (*(volatile struct st_hudi *)0xFFFF8A50) /* H-UDI Address */
```

SH7144F Group
On-Chip Interface I^2C Bus Interface Volume
Application Note

Publication Date: 1st Edition, July 10, 2003
Published by: Sales Strategic Planning Div.
Renesas Technology Corp.
Edited by: Technical Documentation & Information Department
Renesas Kodaira Semiconductor Co., Ltd.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

http://www.renesas.com

SH7144F Group
On-Chip Interface I^{2}C Bus Interface Volume
Application Note

Renesas Electronics Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ05B0080-0100O