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.

H8/300L Series

Interfacing the H8/3644 with an EEPROM

Introduction

Figure 1.1 shows the one-to-one interface between the H8/3644 and EEPROM (HN58X2408) which are connected via a serial clock line (SCL) and a serial data line (SDA).

Target Device

H8/3644

Contents

1. Specifications

1. Figure 1.1 shows the one-to-one interface between the H8/3644 and EEPROM (HN58X2408) which are connected via a serial clock line (SCL) and a serial data line (SDA).
2. Data in ROM is written to EEPROM, and the data written in the EEPROM is read in RAM again.
3. Data to be transferred is a program for turning on an LED connected to the port 7_3.
4. Data is transferred in LSB first.

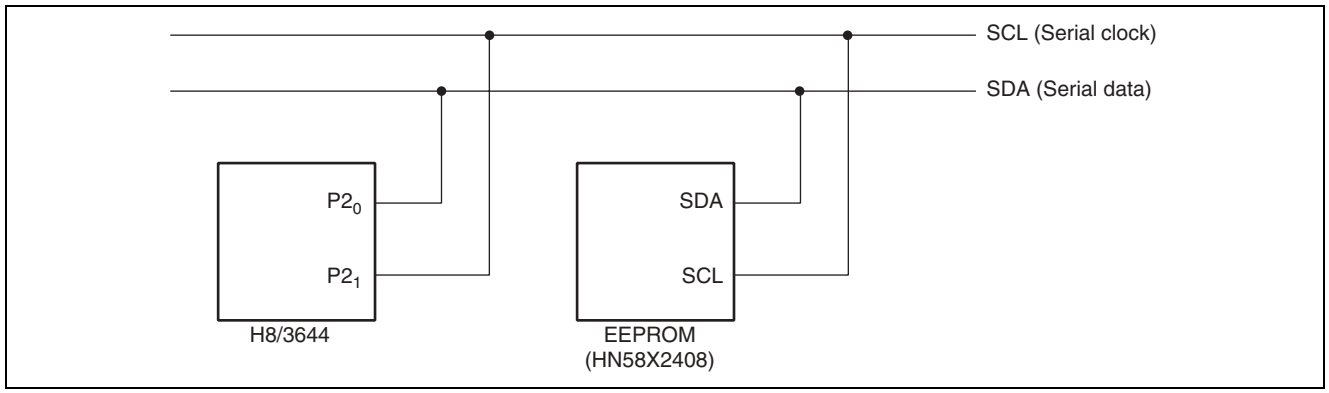


Figure 1.1 Connection between the H8/3644 and EEPROM

2. Concept

1. Figure 2.1 shows EEPROM (HN58X2408) bus timing to be used in this sample task.

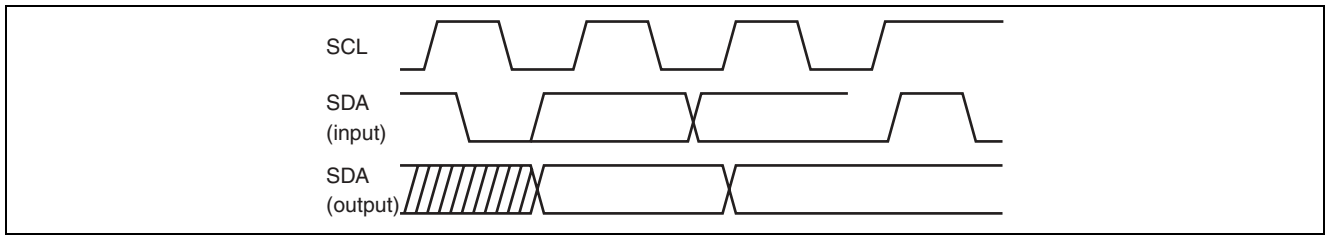


Figure 2.1 EEPROM (HN58X2408) Bus Timing

2. In this sample task, a serial clock to be output from the SCL is generated by setting the $P2_1$ pin level to either high or low through software processing with the bus timings for EEPROM to be used as shown in figure 2.2. The h8/3644 interfaces with the EEPROM (HN58X2408) by outputting/inputting serial data from the pin $P2_0$ synchronized with the serial clock generated through software processing. Figure 2.2 shows pins $P2_1$ and $P2_0$.timing waveforms.

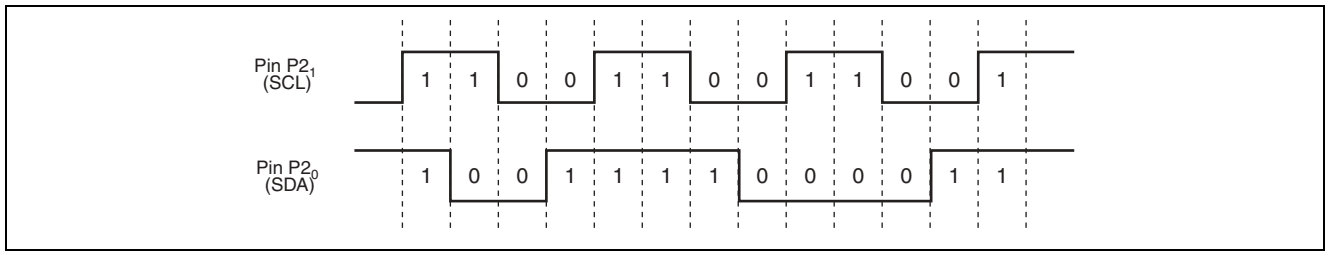


Figure 2.2 Pins $P2_1$ and $P2_0$ Timing Waveforms

3. Description of Functions

1. In this sample task, the H8/3644 interfaces with EEPROM (HN58X2408) through connection as shown in figure 3.1. Table 3.1 shows the pin description of EEPROM (HN58X2408).

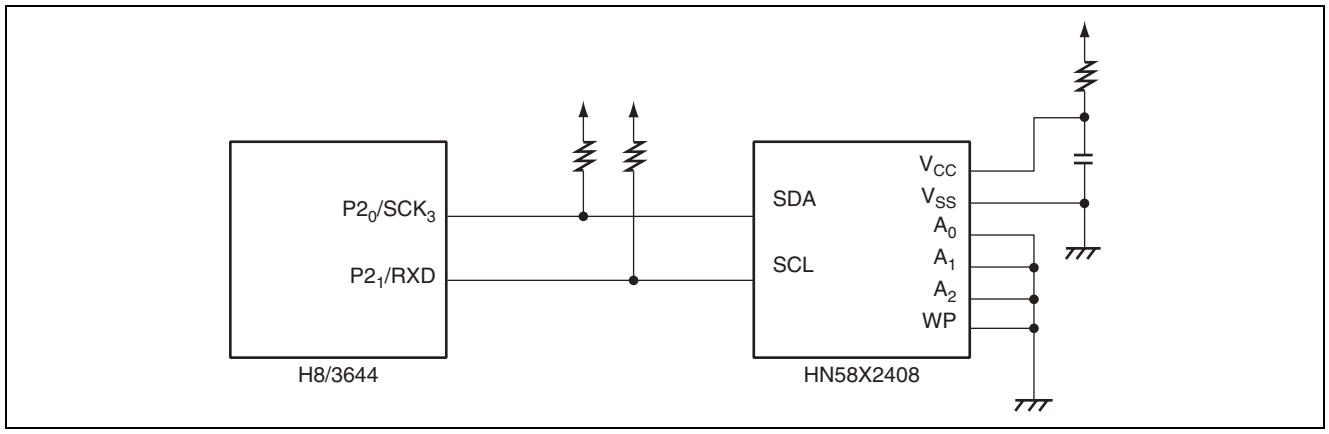


Figure 3.1 Connection between the H8/3644 and HN58X2408

Table 3.1 HN58X2408 Pin Description

Pin Name	Description
A0 to A2	Device address
SCL	Serial clock input
SDA	Serial data input/output
WP	Write protect
V_{CC}	Vcc
V_{SS}	GND

2. The H8/3644 interfaces with EEPROM (HN58X2408) with the configuration of the block diagram as shown in figure 3.2. The functions of the H8/3644 are described below.
   A. The port 2 is a three-bit input/output port, which has three pins, P2$_0$/SCK$_3$, P2$_1$/RXD, and P2$_2$/TXD.
   B. The pin P2$_0$/SCK$_3$ is used as a serial data input/output pin by connecting to the SDA pin of the HN58X2048.
   C. The pin P2$_1$/RXD is used as a serial clock output pin by connecting to the SCL pin of the HN58X2048.
   D. The port 7 is a five-bit input/output port, which has five pins, P7$_3$, P7$_4$/TMRIV, P7$_5$/TCIV, P7$_6$/TMOV, and P7$_7$.
   E. The pin P7$_3$ is used as an output pin to the LED.
   F. The pin P2$_0$/SCK$_3$ functions as a P2$_0$ input/output pin by setting the clock enable 1 (CKE1) and the clock enable 0 in the serial control register 3 (SCR3) and the communication mode (COM) in the serial mode register (SMR) to all 0. The pin P2$_0$/SCK$_3$ functions as a P2$_0$ input pin when the port control register 2$_0$ (PCR2$_0$) in the port control register 2 (PCR2) is set to 0, or functions as a P2$_0$ output pin when PCR2$_0$ is set to 1.
   G. The pin P2$_1$/RXD functions as a P2$_1$ input/output pin by setting the receive enable (RE) in SCR3 to 0. The pin P2$_1$/RXD functions as a P2$_1$ input pin when the port control register 2$_1$ (PCR2$_1$) in PCR2 is set to 0, or functions as a P2$_1$ output pin when PCR2$_1$ is set to 1.
   H. The pin P7$_3$ functions as a P7$_3$ output pin by setting the port control register 7$_3$ (PCR7$_3$) in the port control register 7 (PCR7) to 1.

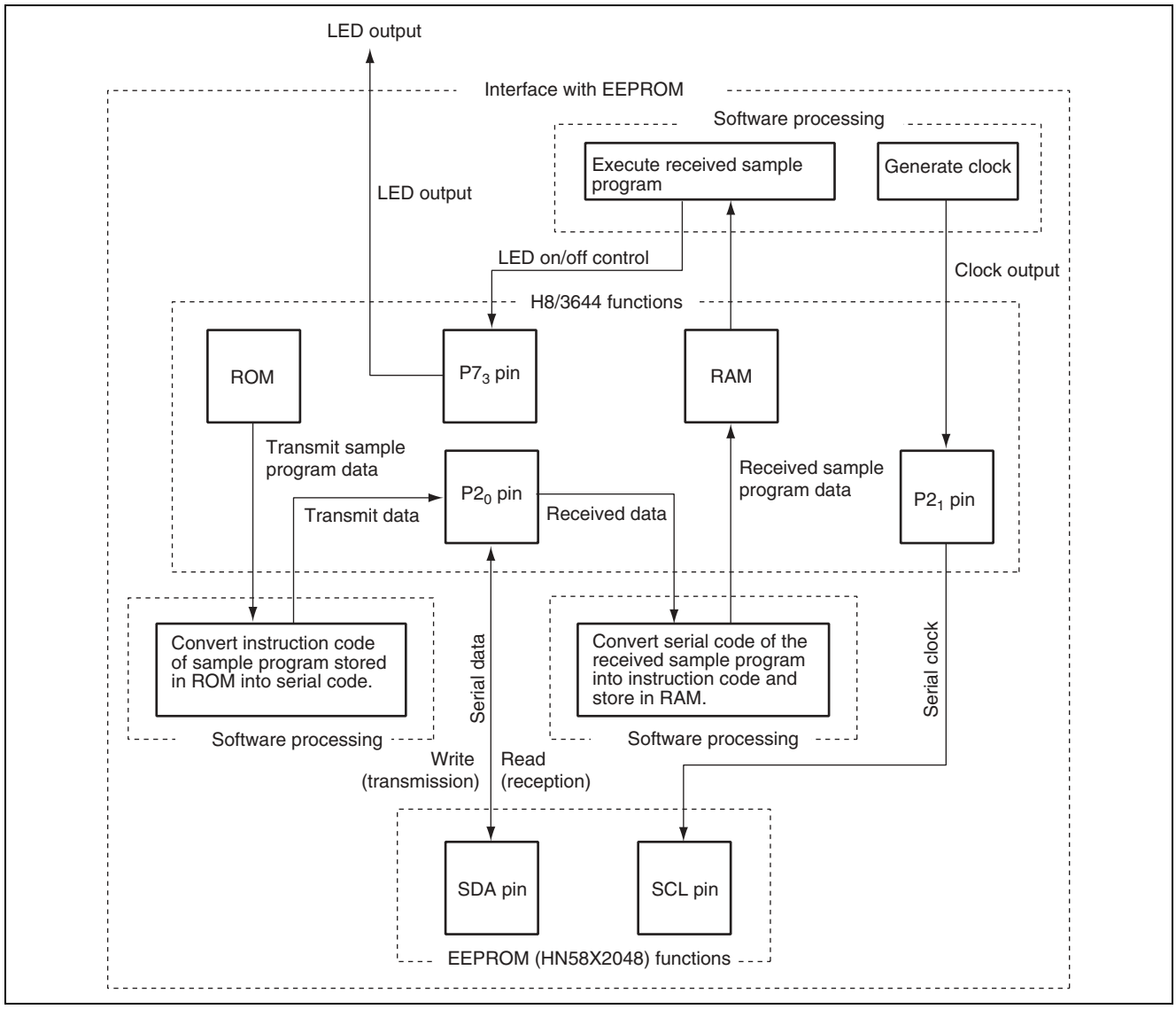


Figure 3.2 Block Diagram of Interface with EEPROM (HN58X2048)

3. Table 3.2 shows function allocations of this sample task. The H8/3644 interfaces with EEPROM through the H8/3644 function allocations as shown in table 3.2.

Table 3.2 H8/3644 Function Allocation

Function	Function Allocation
Pin $P2_0$	Inputs/outputs serial data
$P2_0$	Stores data from the pin $P2_0$
$PCR2_0$	Sets $P2_0$ input/output pin functions
Pin $P2_1$	Serial clock output
$P2_1$	Stores data from the pin $P2_1$
$PCR2_1$	Sets $P2_1$ input/output pin functions
Pin $P7_3$	Outputs LED
$P7_3$	Stores data from the pin $P7_3$
$PCR7_3$	Sets $P7_3$ input/output pin functions

4. Specifications of the EEPROM (HN58X2048) used in this sample task are described below.
   A. The EEPROM (HN58X2048) used in this sample task is two-wired serial interface electrically rewritable ROM (EEPROM), which employs the latest NMOS memory technology, CMOS process, and low-voltage circuit technology and allows operation at low power supply voltage, low power consumption, high-speed operation, and high reliability. High-speed data rewriting is performed through a 32-byte page rewriting function.
   B. Features of the EEPROM (HN58X2048) used in this sample task are shown below.
   — Single power supply: 1.8 to 5.5 V
   — Two-wired serial interface
   — Operating frequency: 400 kHz
   — Current consumption
      - During standby: 3 μA (max.)
      - During reading: 1 mA (max.)
      - During rewriting: 3 mA (max.)
   — Page rewriting: 32-byte page size
   — Rewriting time: 10 ms (2.7 V or greater)/15 ms (1.8 to 2.7 V)
   — Number of rewriting times: 10^5 (during page rewriting)
   C. For starting read/write operation, a start condition must be set by switching the SDA input from high to low during a SCL input high. For stopping read/write operation, a stop condition must be set by switching the SDA input from low to high during a SCL input high. In the case of read operation, when the stop condition is input, EEPROM ends read operation and enters standby state. In the case of write operation, when the stop condition is input, input cycle of data to be written ends, EEPROM enters standby state once data is written in memory during a rewrite cycle (t_{wc}). Figure 3.3 shows start condition/stop condition set timings.

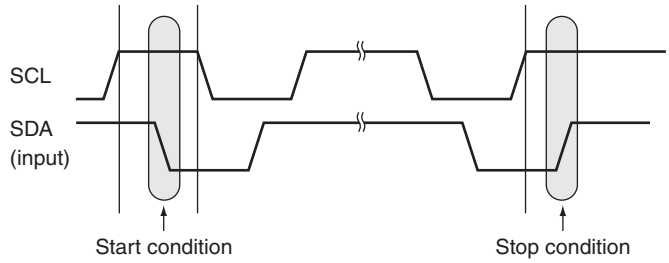


Figure 3.3 Start Condition/Stop Condition Set Timings

D. Serial data such as address information and read information is transferred in 8-bit units. The acknowledge signal is used to indicate that this 8-bit data has been received or transmitted successfully. The reception side outputs 0 at the ninth SCL clock. The transmission side releases the bus mastership to receive the acknowledge signal at the ninth clock. During write operation, the EEPROM receives data, that is, the EEPROM outputs 0, that is, the acknowledge signal at the ninth clock when all 8-bit data have been received. During read operation, the EEPROM outputs 0 of the acknowledge signal after receiving 8-bit data following the start condition. Following this, the EEPROM outputs read data in 8-bit units, then releases the bus mastership, and waits for being transferred 0 of the acknowledge signal. When 0 of the acknowledge signal is detected, the EEPROM outputs data to be read at the next address. When 0 of the acknowledge signal is not detected and the stop condition is received, the EEPROM ends read operation and enters standby state. Note that when 0 of the acknowledge signal is not detected and the stop condition is not transmitted, the EEPROM does not output any data and keeps the bus mastership release state. Figure 3.4 shows the acknowledge signal output timings.

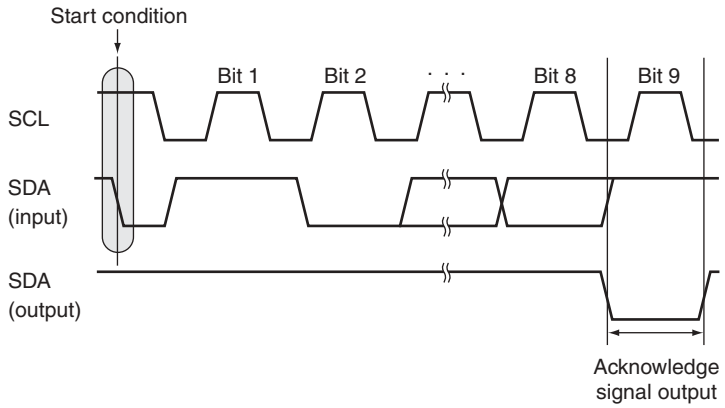


Figure 3.4 Acknowledge Signal Output Timings

E. An 8-bit device address word is input following the starting condition signal to initiate read/write operation of a device. The device address words include a 4-bit device code, a 3-bit device address code, and a 1-bit read/write code. Upper 4 bits of the device address words are device codes used for identifying a device type, which are fixed to '1010' in the EEPROM (HN58X2408) used in this sample task. Following the device codes, the 3-bit device address code is input in order of A2, A1, and A0. The device address codes are used for selecting a device from up to 8 devices connected to the bus. The device address codes in the EEPROM (HN58X2408) used in this sample task are set to '000'. The eighth bit of the device address words is a R/W code, which is used to switch read/write operation. When 0 is input, the EEPROM performs write operation, or when 1 is input, the EEPROM performs read operation. Note that when the device codes are not set to '1010', or when the device address codes do not match, the EEPROM does not perform read/write operation, but enters standby mode. Figure 3.5 shows the device address words.

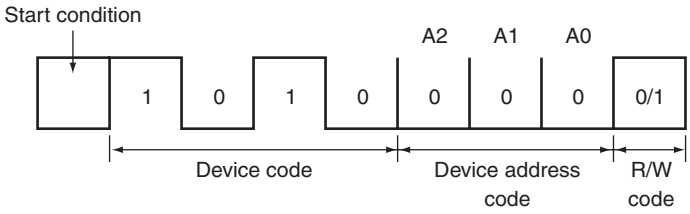


Figure 3.5 Device Address Words

F. In this sample task, a page-write function for rewriting an arbitrary number of bytes up to 32 bytes at once is used to perform write operation. The start condition, the device address words, the memory addresses (n), and write data (Dn) are input in that order while confirming 0 output of the acknowledge signal for each of 9 bits. When write data (Dn + 1) is input, the EEPROM enters page write mode. When the write data (Dn + 1) is input, addresses in the page (a0 through a4) are automatically incremented by 1 to an address (n + 1). In this manner, write data can be input sequentially, addresses in the page are incremented for each write data input, as a result, up to 32 bytes of write data can be input. When the addresses in the page (a0 through a4) reach the last address in the page, the address is rolled over and is returned to the start address of the page. When roll over occurs, write data is input twice or more to the same address, however, the latest input write data is effective. When the stop condition is input, write data input ends and the EEPROM starts rewrite operation. Figure 3.6 shows the page write operation.

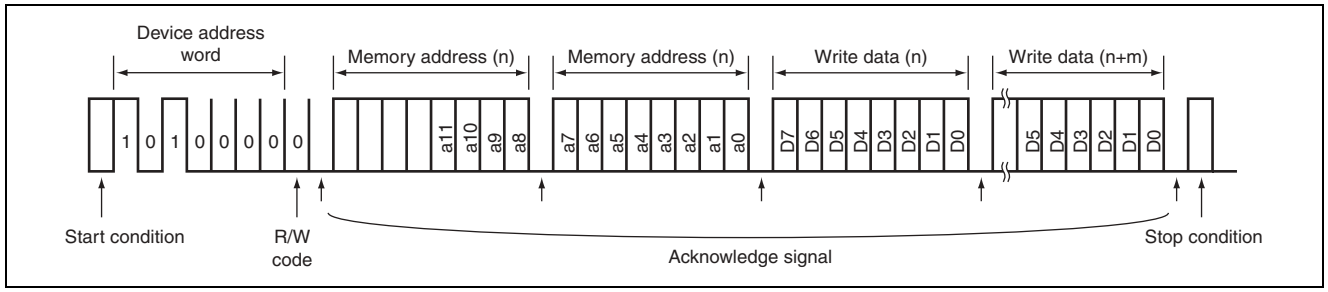


Figure 3.6 Page Write Operation

G. Acknowledge polling is a function to determine whether or not EEPROM is being rewritten. An 8-bit device address word is input following the start condition during rewrite cycle. In the case of the acknowledge polling, read/write code may be set to any of 1 or 0. The acknowledge signal at the ninth bit is used to determine whether or not EEPROM is being rewritten. When the acknowledge signal is 1, EEPROM is being rewritten, or when the acknowledge signal is 0, rewrite cycle ends. The acknowledgement polling starts functioning when the stop condition is input after data to be written is input.

H. In this sample task, the EEPROM performs read operation using sequential read mode for reading data sequentially. To begin with, the start address of data to be read in dummy write mode is input. When 0 of the acknowledge signal is input after outputting 8-bit data, an address is incremented by 1, and the next 8-bit data is output. When 0 of the acknowledge signal is input continuously after outputting data, data is sequentially output while incrementing the address by 1. When address reaches the last address, address is rolled over to address 0. Sequential read is possible after roll over. To stop operation, 1 of the acknowledge signal (alternatively release of the bus mastership without inputting the acknowledge signal) and the stop condition should be input in that order. Figure 3.7 shows the sequential read operation.

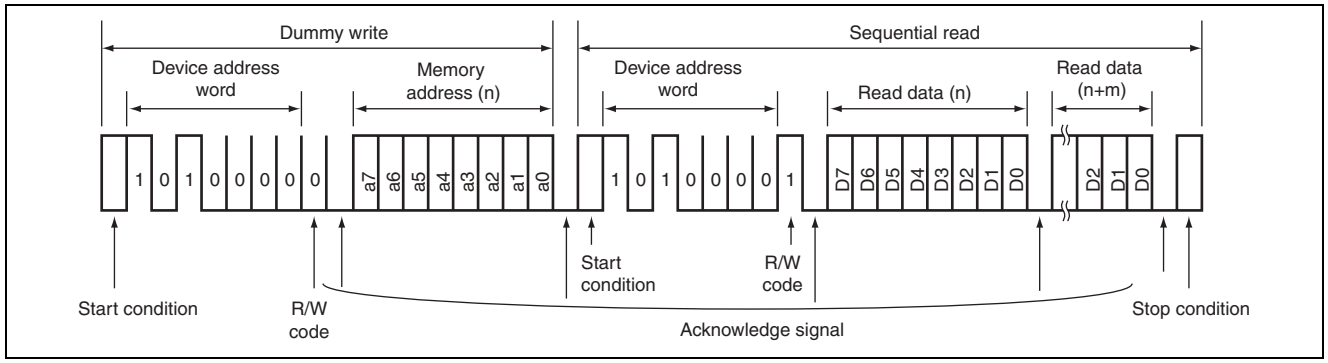


Figure 3.7 Sequential Read Operation

I. EEPROM should wait during rewrite cycle (t_{wc}) for entering read operation after write operations end. The rewrite cycle (t_{wc}) is 10 ms (max.) during 5-V operation. Figure 3.8 shows rewrite cycle timings.

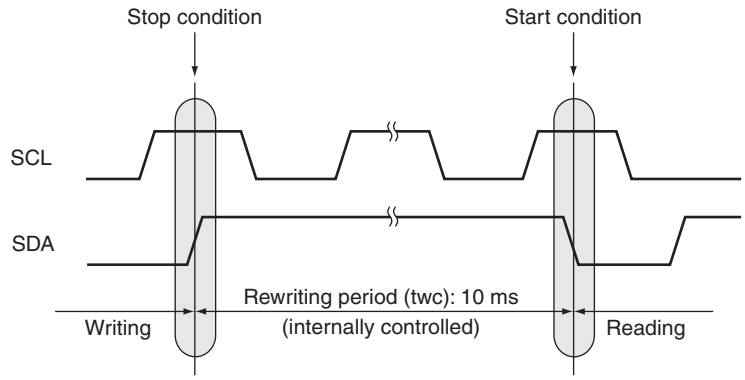


Figure 3.8 Rewrite Cycle Timings

4. Principles of Operation

1. Figure 4.1 illustrates the principle of operation of this sample task. As shown in figure 4.1, data a rewritten (transmitted) to EEPROM through the H8/3644 hardware processing and software processing.

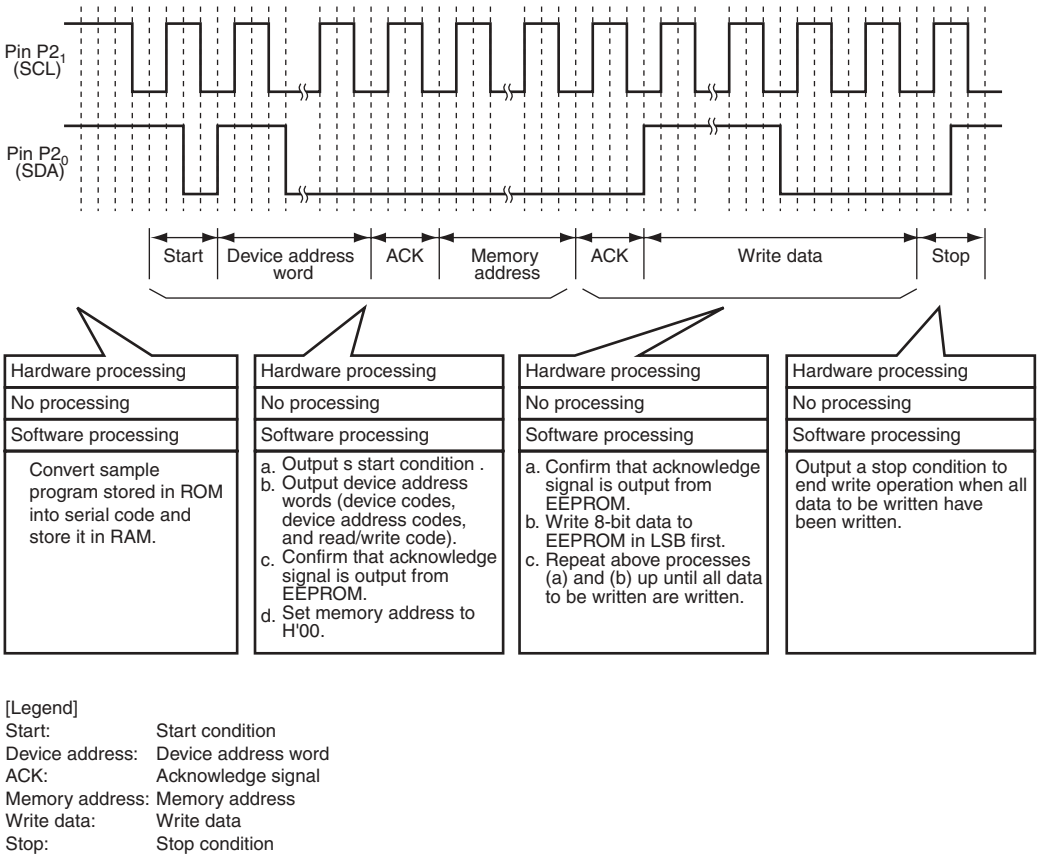


Figure 4.1 Principle of Operation during Writing to EEPROM

2. Figure 4.2 shows the principle of operation during read operation (reception). As shown in figure 4.2, data is read (received) from EEPROM through the H8/3644 hardware/software processing.

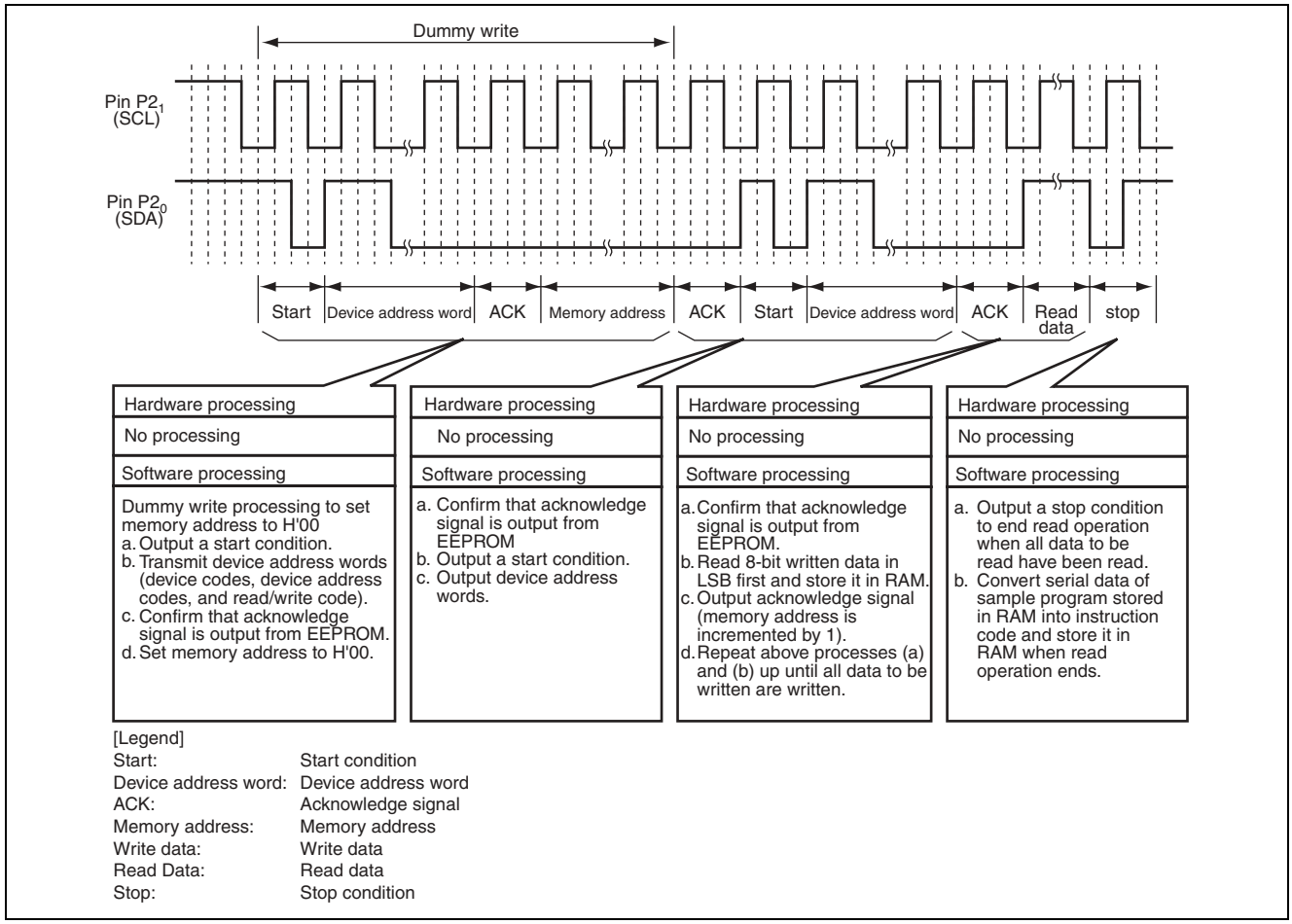


Figure 4.2 Principle of Operation during Reading from EEPROM

3. Table 4.1 shows input/output operation to/from the port 2 used in this sample task. Through settings as shown in table 4.1, the serial clock is output and serial data is input/output.

Table 4.1 $P2_1$ (SCL) and $P2_0$ (SDA) Input/Output Settings

Pin setting \ Output value		$P2_1$ (SCL) = 1 $P2_0$ (SDA) = 1	$P2_1$ (SCL) = 1 $P2_0$ (SDA) = 0	$P2_1$ (SCL) = 0 $P2_0$ (SDA) = 1	$P2_1$ (SCL) = 0 $P2_0$ (SDA) = 0
PDR2	$P2_1$	0	0	0	0
	$P2_0$	0	0	0	0
PCR2	$PCR2_1$	0 (input pin function)	0 (input pin function)	1 (output pin function)	1 (output pin function)
	$PCR2_0$	0 (input pin function)	1 (output pin function)	0 (input pin function)	1 (output pin function)

4. Figure 4.3 shows the H8/3644 memory map used in this sample task.

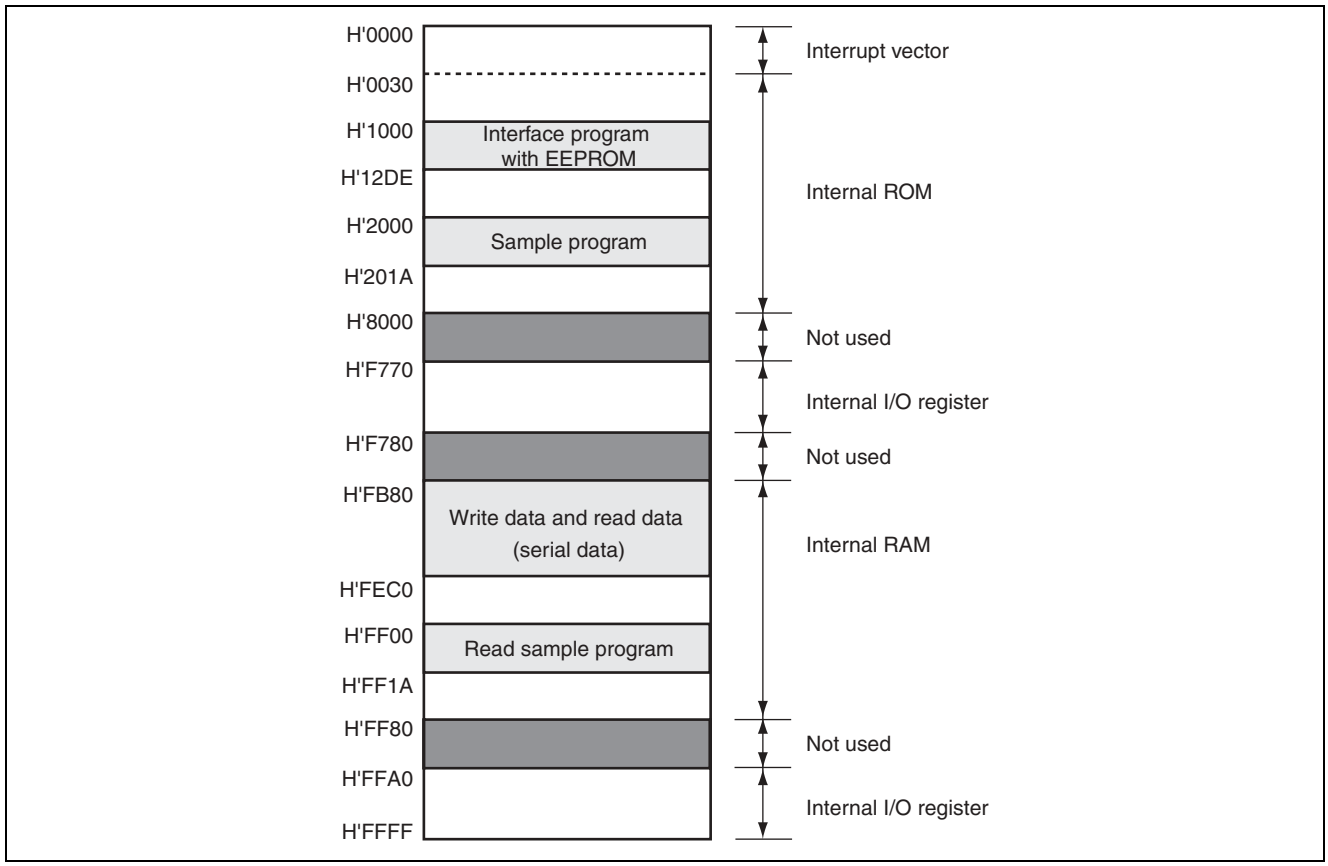


Figure 4.3 H8/3644 Memory Map used in this Sample Task

5. Description of Software

5.1 Modules

Table 5.1 describes the modules used in this sample task.

Table 5.1 Description of Modules

Module	Label	Function
Main routine	MAIN	Initializes stack pointer, disables interrupt, initializes RAM to be used, waits during rewrite cycle, and controls read/write from/to EEPROM
Serial code conversion	SERCODE	Converts sample program data in ROM into serial data and stores it in RAM
Write	WRITE	Transmits serial data of sample program stored in RAM to EEPROM in LSB first
Read	READ	Receives serial data of sample program from EEPROM in LSB first and stores it in RAM
Instruction code conversion	PARCODE	Converts serial data of sample program stored in RAM into instruction code
Received sample program	SPLPGM	Sample program received from EEPROM that repeats turning the LED connected to pin $P7_3$ on or off for each of 262 ms
Error routine	ERROR	Performs error processing

5.2 Arguments

This sample task does not use arguments.

5.3 Internal Registers

The H8/3644 internal registers used in this sample task are described in table 5.2.

Table 5.2 Description of the H8/3644 Internal Registers

Register		Function	Address	Setting
PDR2	$P2_1$	Port data register 2 (port data register 2_1) when $P2_1$ is 0, $P2_1$ pin data is 0 when $P2_1$ is 1, $P2_1$ pin data is 1	H'FFD5 Bit 1	0/1
	$P2_0$	Port data register 2 (port data register 2_0) when $P2_0$ is 0, $P2_0$ pin data is 0 when $P2_0$ is 1, $P2_0$ pin data is 1	H'FFD5 Bit 0	0/1
PCR2	$PCR2_1$	Port control register 2 (port control register 2_1) when $PCR2_1$ is 0, $P2_1$ pin functions as an input pin when $PCR2_1$ is 1, $P2_1$ pin functions as an output pin	H'FFE5 Bit 1	0/1
	$PCR2_0$	Port control register 2 (port control register 2_0) when $PCR2_0$ is 0, $P2_0$ pin functions as an input pin when $PCR2_0$ is 1, $P2_0$ pin functions as an output pin	H'FFE5 Bit 0	0/1
PDR7	$P7_3$	Port data register 7 (port data register 7_3) when $P7_3$ is 0, $P7_3$ pin data is 0 when $P7_3$ is 1, $P7_3$ pin data is 1	H'FFDA Bit 3	0/1
PCR7	$PCR7_3$	Port control register 7 (port control register 7_3) when $PCR7_3$ is 0, $P7_3$ pin functions as an input pin when $PCR7_3$ is 1, $P7_3$ pin functions as an output pin	H'FFEA Bit 3	1

5.4 Description of RAM

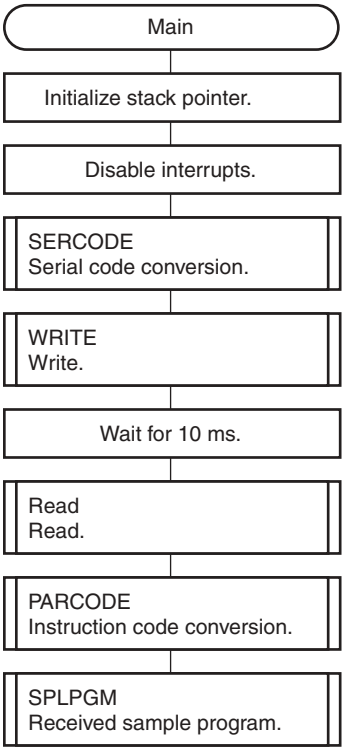
Table 5.3 describes the RAM used in this sample task.

Table 5.3 Description of RAM

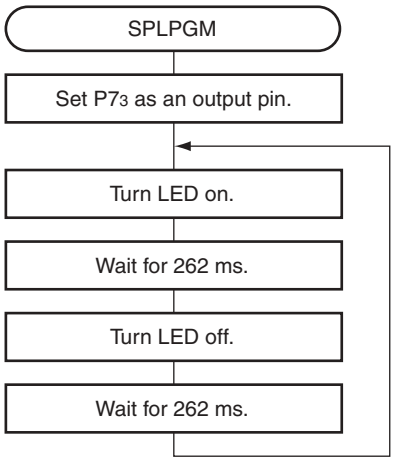
Label	Function	Address	Used in
SERAREA	Stores the start address of RAM to which sample program serial data is stored	H'FB80	Serial code conversion Write Read Instruction code conversion
COUNTER	Stores the last address of RAM to which sample program serial data is stored	H'FEC0	Serial code conversion
SPLPGM	Stores the start address of RAM to which sample program instruction code read from EEPROM is stored	H'FF00	Serial code conversion Instruction code conversion

6. Flowchart

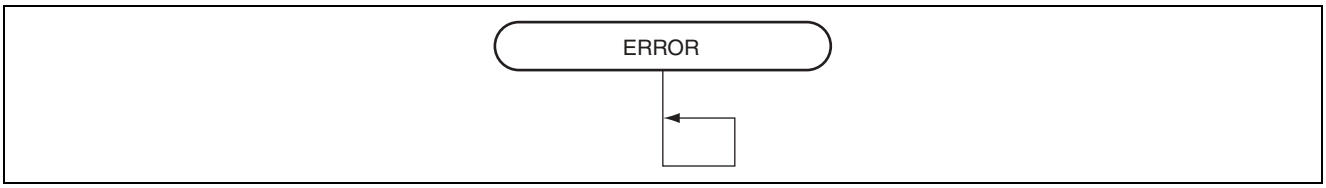
1. Main routine



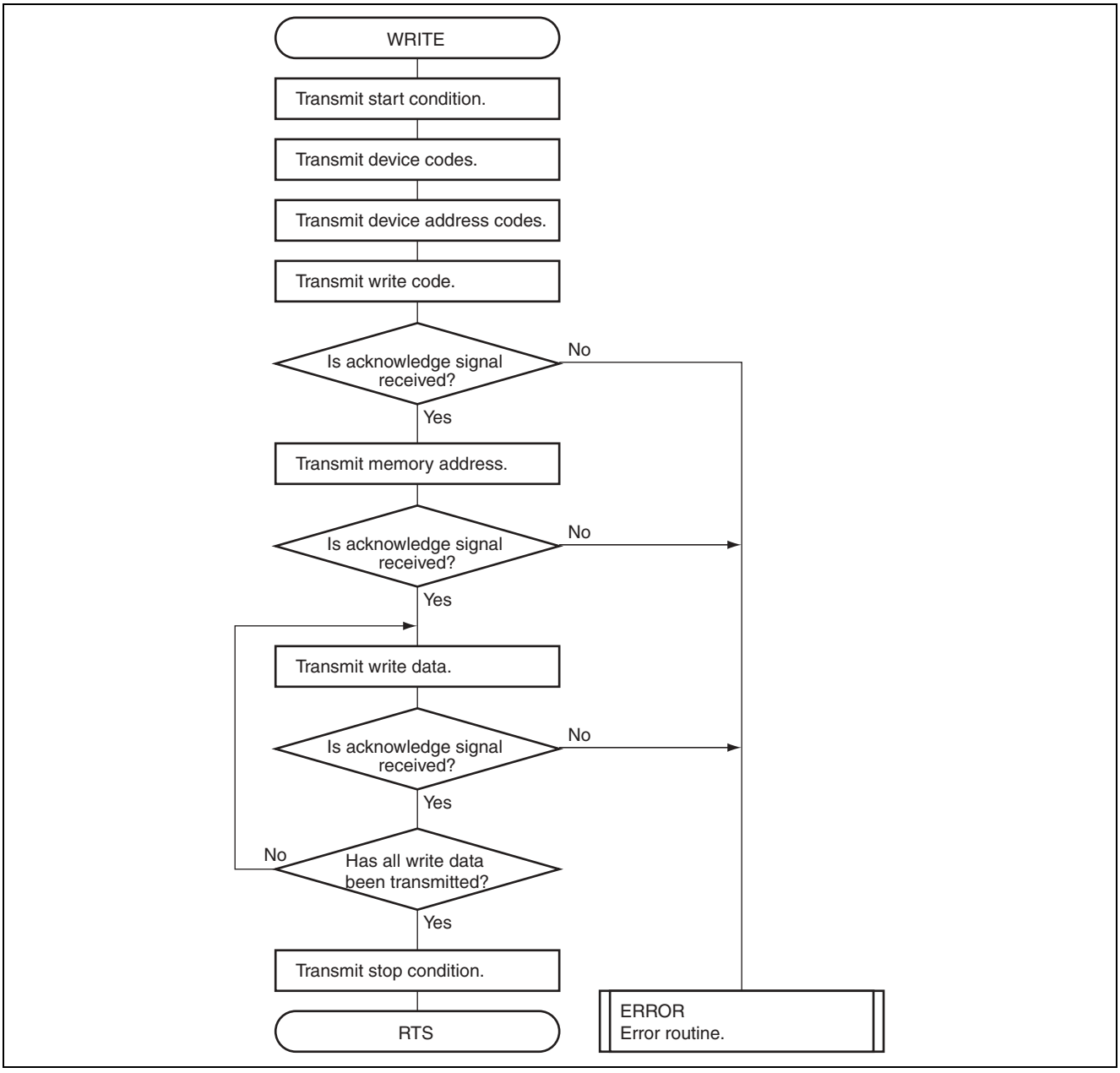
2. Received sample program



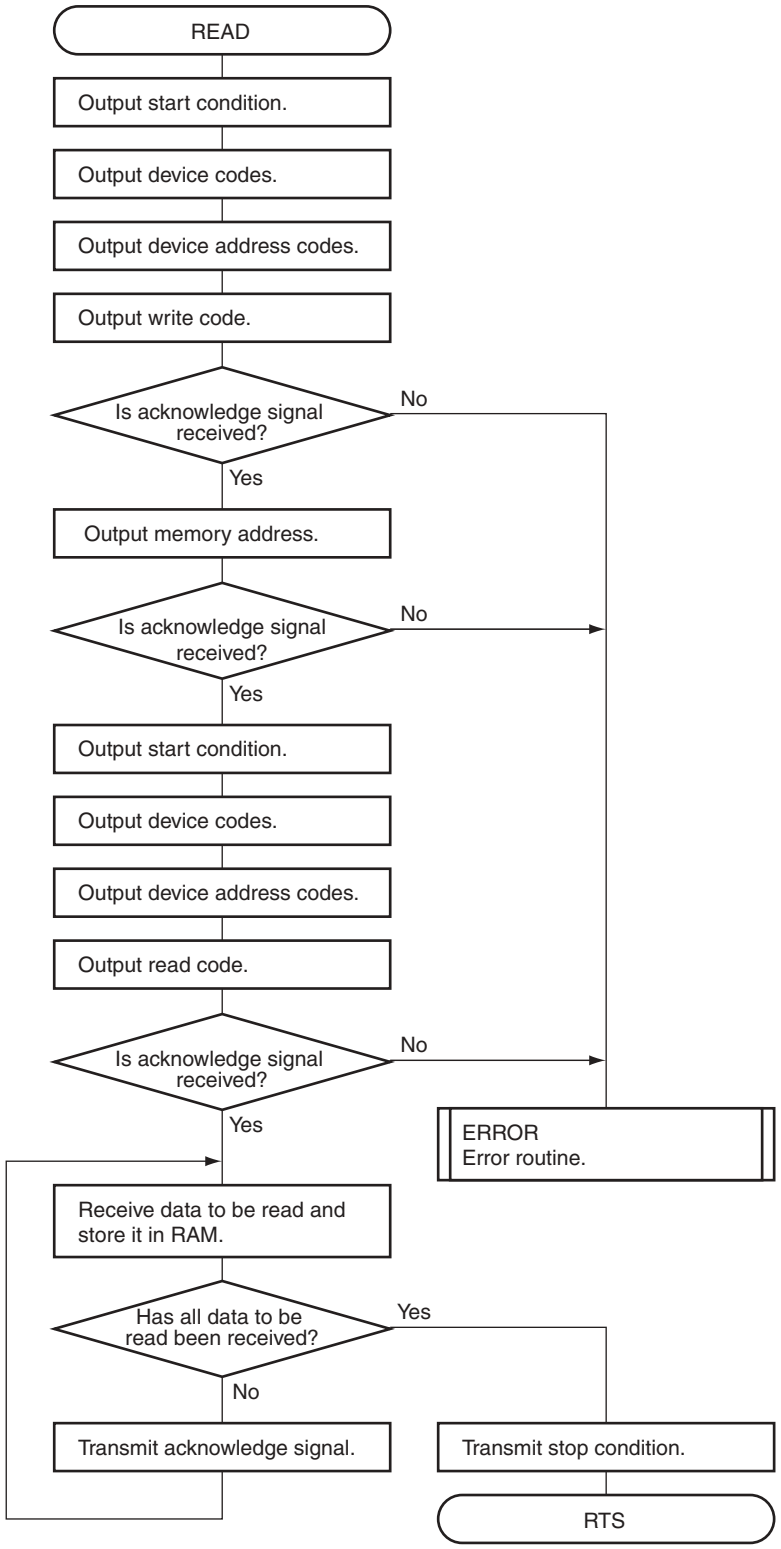
3. Error routine



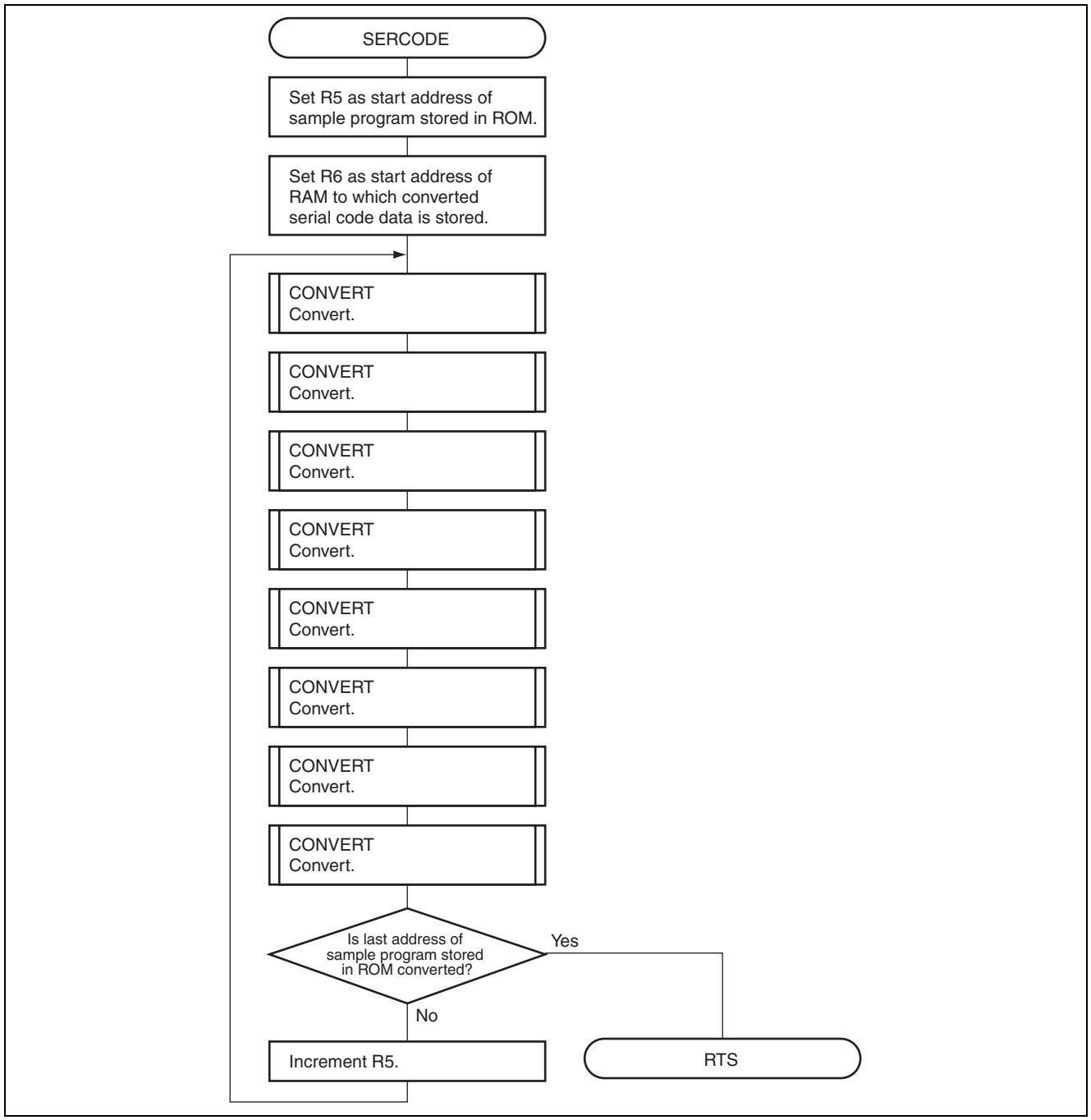
4. Write



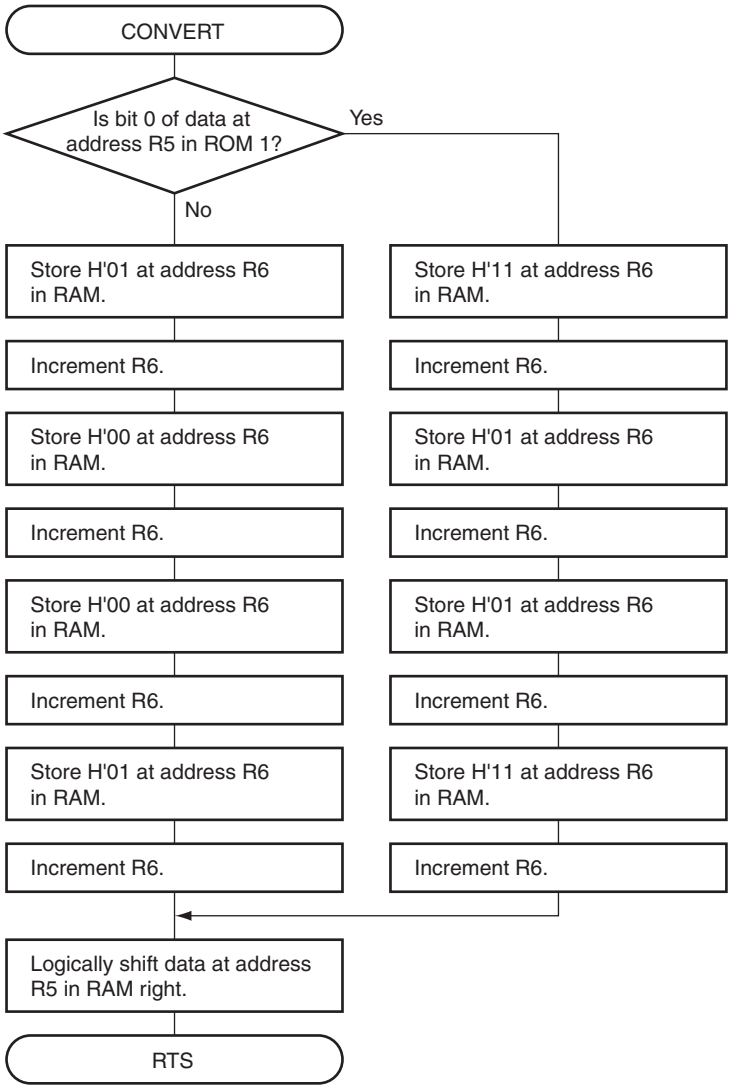
5. Read


READ
Output start condition.
Output device codes.
Output device address codes.
Output write code.
Is acknowledge signal received?
No
Yes
Output memory address.
Is acknowledge signal received?
No
Yes
Output start condition.
Output device codes.
Output device address codes.
Output read code.
Is acknowledge signal received?
No
Yes
ERROR
Error routine.
Receive data to be read and store it in RAM.
Has all data to be read been received?
Yes
No
Transmit acknowledge signal.
Transmit stop condition.
RTS

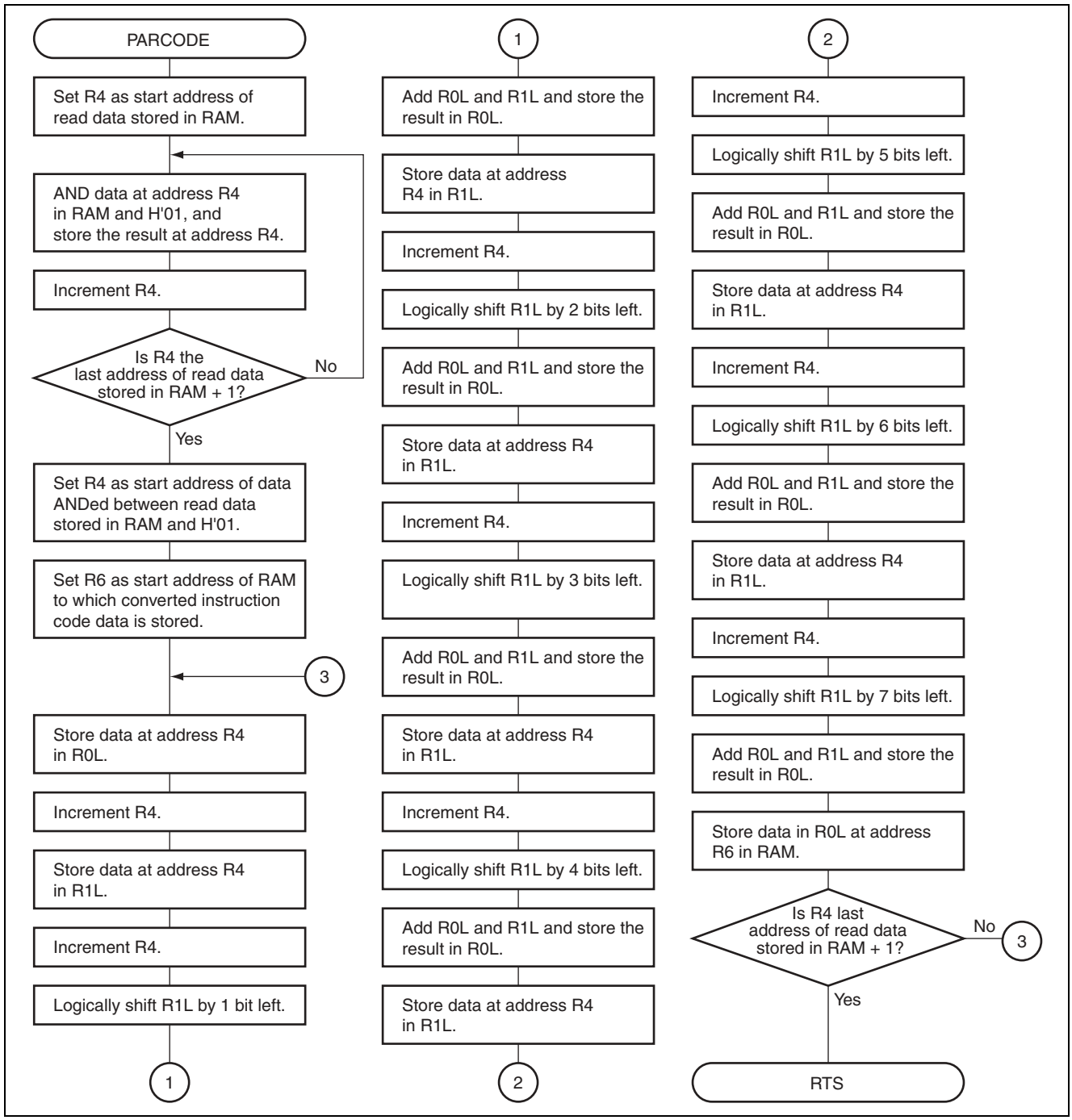
6. Serial code conversion



7. Conversion



8. Instruction code conversion



7. Program Listing

```
;*************************************************************************************
;*                                                                                   *
;*     H8/3644 Application Note                                                      *
;*                                                                                   *
;*     'EEPROM Write & Read Control'                                                 *
;*                                                                                   *
;*     Function : I/O Port Base                                                      *
;*                                                                                   *
;*     External Clock : 10MHz                                                        *
;*     Internal Clock :  5MHz                                                        *
;*                                                                                   *
;*************************************************************************************
;
          .cpu         300L
;
;*************************************************************************************
;*  Symbol Defnition                                                                 *
;*************************************************************************************
;
PDR2      .equ         H'FFD5        ;Port Data Register 2
PCR2      .equ         H'FFE5        ;Port Control Register 2
;*************************************************************************************
;*  Ram Allocation                                                                   *
;*************************************************************************************
;
SERAREA   .equ         H'FB80        ;Serial code area (H'FB80-H'FEBF: SERSIZE * 4 pulse)
COUNTER   .equ         H'FEC0        ;Counter (@H'FEC0 = H'1A: 26 bytes)
SPLPGM    .equ         H'FF00        ;Sample program area (H'FF00-H'FF19)
STACK     .equ         H'FF80        ;Stack Pointer
SPLSIZE   .equ         H'1A          ;Sample program size ( = 26 bytes)
SERSIZE   .equ         H'D0          ;Serial code size ( = SPLSIZE * 8-bit)
;*************************************************************************************
;*  Vector Address                                                                   *
;*************************************************************************************
;
          .org         H'0000
          .data.w      MAIN          ;Reset Interrupt
          .org         H'0008
          .data.w      MAIN          ;IRQ0 Interrupt
          .data.w      MAIN          ;IRQ1 Interrupt
          .data.w      MAIN          ;IRQ2 Interrupt
          .data.w      MAIN          ;IRQ3 Interrupt
          .data.w      MAIN          ;INT0 - INT7 Interrupt
          .data.w      H'0014
          .data.w      MAIN          ;Timer A Interrupt
          .data.w      MAIN          ;Timer B1 Interrupt
          .data.w      H'0020
          .data.w      MAIN          ;Timer X Interrupt
          .data.w      MAIN          ;Timer V Interrupt
          .org         H'0026
          .data.w      MAIN          ;Sci1 Interrupt
          .org         H'002A
          .data.w      MAIN          ;Sci3 Interrupt
          .data.w      MAIN          ;A/D Converter Interrupt
          .data.w      MAIN          ;Sleep Interrupt
```

```
;*************************************************************************************
;*  Main Program                                                                     *
;*************************************************************************************
;
          .org          H'1000
MAIN      MOV.W         #STACK,SP      ;Initialize Stack Pointer
          ORC           #H'80,CCR      ;Interrupt Disable
          BSR           SERCODE        ;Convert sample program data to serial code
          JSR           @WRITE         ;Write EEPROM
          MOV.W         #1,R4          ;10ms wait as tWC spec. of EEPROM
          MOV.W         #H'208C,R5
TWCWAIT   SUB.W         R4,R5
          BNE           TWCWAIT
          JSR           @READ          ;Read EEPROM
          JSR           @PARCODE       ;Convert Serial code to 8-bit sample program data
          MOV.W         #1,R0          ;Load software time data of LED on/off period
          MOV.W         #0,R1          ;as 262ms
          MOV.W         #1,R2
          JMP           @SPLPGM        ;Execute sample program at Internal RAM
;
ERROR     BRA           ERROR          ;ERROR area
;*************************************************************************************
;  Convert sample program to serial code                                            *
;*************************************************************************************
;
SERCODE   .equ          $
          MOV.B         #0,R0L         ;Load SCL = 1, SDA = 1 data
          MOV.B         #1,R1L         ;Load SCL = 1, SDA = 0 data
          MOV.B         #2,R2L         ;Load SCL = 0, SDA = 1 data
          MOV.B         #3,R3L         ;Load SCL = 0, SDA = 0 data
          MOV.W         #SAMPLE,R5     ;Load sample program address
          MOV.W         #SERAREA,R6    ;Load serial code address
          MOV.B         #SPLSIZE,R4L   ;Load sample program size
          MOV.B         R4L,@COUNTER
NEXTCON   MOV.B         @R5+,R4L       ;Load sample program data
          BSR           CONVERT        ;Convert sample program data to serial code bit0
          BSR           CONVERT        ;Convert sample program data to serial code bit1
          BSR           CONVERT        ;Convert sample program data to serial code bit2
          BSR           CONVERT        ;Convert sample program data to serial code bit3
          BSR           CONVERT        ;Convert sample program data to serial code bit4
          BSR           CONVERT        ;Convert sample program data to serial code bit5
          BSR           CONVERT        ;Convert sample program data to serial code bit6
          BSR           CONVERT        ;Convert sample program data to serial code bit7
          MOV.B         @COUNTER,R4L
          DEC           R4L
          MOV.B         R4L,@COUNTER
          CMP.B         #0,R4L         ;@COUNTER=0?
          BNE           NEXTCON        ;No.
          RTS
;
CONVERT   .equ          $
          BTST          #0,R4L         ;If sample program data bitn is "0",
          BEQ           BITEQU0        ;it branch to BITEQU0
          MOV.B         R2L,@R6        ;Store SCL & SDA data as follows.
          ADDS          #1,R6          ;SCL = 0110
          MOV.B         R0L,@R6        ;SDA = 1111
          ADDS          #1,R6
          MOV.B         R0L,@R6
          ADDS          #1,R6
```

```
          MOV.B       R2L,@R6
          ADDS        #1,R6
          BRA         BITn
BITEQU0   MOV.B       R3L,@R6       ;Store SCL & SDA data as follows.
          ADDS        #1,R6         ;SCL = 0110
          MOV.B       R1L,@R6       ;SDA = 0000
          ADDS        #1,R6
          MOV.B       R1L,@R6
          ADDS        #1,R6
          MOV.B       R3L,@R6
          ADDS        #1,R6
BITn      SHLR        R4L
          RTS
;*************************************************************************************
;*  Write (Use Page Write Operation)                                                 *
;*************************************************************************************
;
WRITE     .equ        $
          MOV.B       R0L,@PDR2     ;SCL = "0", SDA = "1"
          MOV.B       R2L,@PCR2
;
          JSR         @RW_start     ;Output "0" of Device address start bit
          JSR         @RW_H         ;Output "1" of Device address
          JSR         @RW_L         ;Output "0" of Device address
          JSR         @RW_H         ;Output "1" of Device address
          JSR         @RW_L         ;Output "0" of Device address
          JSR         @RW_L         ;Output "0" of Device address code
          JSR         @RW_L         ;Output "0" of Device address code
          JSR         @RW_L         ;Output "0" of Device address code
          JSR         @RW_L         ;Output "0" of Device address write bit
          JSR         @RW_ack       ;Input "0" of /ACK
;
          JSR         @RW_L         ;Output "0" of Memory address a7
          JSR         @RW_L         ;Output "0" of Memory address a6
          JSR         @RW_L         ;Output "0" of Memory address a5
          JSR         @RW_L         ;Output "0" of Memory address a4
          JSR         @RW_L         ;Output "0" of Memory address a3
          JSR         @RW_L         ;Output "0" of Memory address a2
          JSR         @RW_L         ;Output "0" of Memory address a1
          JSR         @RW_L         ;Output "0" of Memory address a0
          JSR         @RW_ack       ;Input "0" of /ACK
;
          MOV.W       #SERAREA,R4   ;Load serial code of sample program
          MOV.W       #SPLSIZE,R5
WRLOOP    JSR         @WR_data      ;Output Write data D0
          JSR         @WR_data      ;Output Write data D1
          JSR         @WR_data      ;Output Write data D2
          JSR         @WR_data      ;Output Write data D3
          JSR         @WR_data      ;Output Write data D4
          JSR         @WR_data      ;Output Write data D5
          JSR         @WR_data      ;Output Write data D6
          JSR         @WR_data      ;Output Write data D7
          JSR         @RW_ack       ;Input "0" of /ACK
          DEC         R5L           ;Counter=0?
          BEQ         WREND         ;No.
          BRA         WRLOOP
WREND     JSR         @RW_stop      ;Output stop bit
          RTS
```

```
;*******************************************************************************
;*  Read (Random & Sequential Operation of EEPROM)                             *
;*******************************************************************************
;
READ      .equ          $
          MOV.W         #SERAREA,R4    ;Load serial code address
          MOV.W         #SPLSIZE,R5    ;Load count data ( = 26 bytes)
          MOV.B         #0,R0L         ;Load SCL = 1, SDA = 1 data
          MOV.B         #1,R1L         ;Load SCL = 1, SDA = 0 data
          MOV.B         #2,R2L         ;Load SCL = 0, SDA = 1 data
          MOV.B         #3,R3L         ;Load SCL = 0, SDA = 0 data
          MOV.B         R0L,@PDR2
          MOV.B         R2L,@PCR2      ;SCL = "0", SDA = "1"
;
          JSR           @RW_start      ;Output "0" of Device address start bit
          JSR           @RW_H          ;Output "1" of Device address
          JSR           @RW_L          ;Output "0" of Device address
          JSR           @RW_H          ;Output "1" of Device address
          JSR           @RW_L          ;Output "0" of Device address
          JSR           @RW_L          ;Output "0" of Device address code
          JSR           @RW_L          ;Output "0" of Device address code
          JSR           @RW_L          ;Output "0" of Device address code
          JSR           @RW_L          ;Output "0" of Device address write bit
          JSR           @RW_ack        ;Input "0" of /ACK
;
          BSR           RW_L           ;Output "0" of Memory address a7
          BSR           RW_L           ;Output "0" of Memory address a6
          BSR           RW_L           ;Output "0" of Memory address a5
          BSR           RW_L           ;Output "0" of Memory address a4
          BSR           RW_L           ;Output "0" of Memory address a3
          BSR           RW_L           ;Output "0" of Memory address a2
          BSR           RW_L           ;Output "0" of Memory address a1
          BSR           RW_L           ;Output "0" of Memory address a0
          BSR           RW_ack         ;Input "0" of /ACK
;
          BSR           RW_start       ;Output "0" of Device address start bit
          BSR           RW_H           ;Output "1" of Device address
          BSR           RW_L           ;Output "0" of Device address
          BSR           RW_H           ;Output "1" of Device address
          BSR           RW_L           ;Output "0" of Device address
          BSR           RW_L           ;Output "0" of Device address code
          BSR           RW_L           ;Output "0" of Device address code
          BSR           RW_L           ;Output "0" of Device address code
          BSR           RW_H           ;Output "1" of Device address read bit
          BSR           RW_ack         ;Input "0" of /ACK
;
RDLOOP    JSR           @RD_data       ;Input Read data D0
          JSR           @RD_data       ;Input Read data D1
          JSR           @RD_data       ;Input Read data D2
          JSR           @RD_data       ;Input Read data D3
          JSR           @RD_data       ;Input Read data D4
          JSR           @RD_data       ;Input Read data D5
          JSR           @RD_data       ;Input Read data D6
          JSR           @RD_data       ;Input Read data D7
          BSR           RW_L           ;Output "0" of /ACK bit
          DEC           R5L
          BEQ           SERDEND
          BRA           RDLOOP
```

```
SERDEND   .equ        $
          MOV.B       R0L,@PDR2
          MOV.B       R3L,@PCR2     ;SCL = "0", SDA = "0"
          BSR         RW_stop       ;Output stop bit
          RTS
;*************************************************************************************
;  Subroutine                                                                        *
;*************************************************************************************
;
RW_start  .equ        $
          MOV.B       R0L,@PDR2     ;Output "0" of Start bit
          MOV.B       R2L,@PCR2     ;SCL = "0", SDA = "1"
          MOV.B       R0L,@PDR2
          MOV.B       R0L,@PCR2     ;SCL = "1", SDA = "1"
          MOV.B       R0L,@PDR2
          MOV.B       R1L,@PCR2     ;SCL = "1", SDA = "0"
          MOV.B       R0L,@PDR2
          MOV.B       R3L,@PCR2     ;SCL = "0", SDA = "0"
          RTS
;
RW_H      .equ        $
          MOV.B       R0L,@PDR2     ;Output "1" of Device/Memory address
          MOV.B       R2L,@PCR2     ;SCL = "0", SDA = "1"
          MOV.B       R0L,@PDR2
          MOV.B       R0L,@PCR2     ;SCL = "1", SDA = "1"
          MOV.B       R0L,@PDR2
          MOV.B       R0L,@PCR2     ;SCL = "1", SDA = "1"
          MOV.B       R0L,@PDR2
          MOV.B       R2L,@PCR2     ;SCL = "0", SDA = "1"
          RTS
;
RW_L      .equ        $
          MOV.B       R0L,@PDR2     ;Output "0" of Device/Memory address or /ACK
          MOV.B       R3L,@PCR2     ;SCL = "0", SDA = "0"
          MOV.B       R0L,@PDR2
          MOV.B       R1L,@PCR2     ;SCL = "1", SDA = "0"
          MOV.B       R0L,@PDR2
          MOV.B       R1L,@PCR2     ;SCL = "1", SDA = "0"
          MOV.B         R0L,@PDR2
          MOV.B       R3L,@PCR2     ;SCL = "0", SDA = "0"
          RTS
;
RW_ack   .equ         $
          MOV.B       R0L,@PDR2     ;Input "0" of /ACK bit
          MOV.B       R2L,@PCR2
          MOV.B       R0L,@PDR2
          MOV.B       R0L,@PCR2
          BTST        #0,@PDR2      ;/ACK=0?
          BEQ         ACKOK         ;Yes.
          JMP         @ERROR
ACKOK     MOV.B       R0L,@PDR2
          MOV.B       R3L,@PCR2     ;SCL = "0", SDA = "0"
          RTS
;
```

```
RW_stop   .equ        $
          MOV.B       R0L,@PDR2     ;Output "1" of Stop bit
          MOV.B       R1L,@PCR2     ;SCL = "1", SDA = "0"
          MOV.B       R0L,@PDR2
          MOV.B       R0L,@PCR2     ;SCL = "1", SDA = "1"
          MOV.B       R0L,@PDR2
          MOV.B       R2L,@PCR2     ;SCL = "0", SDA = "1"
          MOV.B       R0L,@PDR2
          MOV.B       R2L,@PCR2     ;SCL = "0", SDA = "1"
          MOV.B       R0L,@PDR2
          MOV.B       R0L,@PCR2     ;SCL = "1", SDA = "1"
          RTS
;
WR_data   .equ        $
          MOV.B       @R4+,R1L      ;Output write data bitn
          MOV.B       R1L,@PCR2     ;SCL = "0", SDA = "Dn Output"
          MOV.B       @R4+,R1L
          MOV.B       R1L,@PCR2     ;SCL = "1", SDA = "Dn Output"
          MOV.B       @R4+,R1L
          MOV.B       R1L,@PCR2     ;SCL = "1", SDA = "Dn Output"
          MOV.B       @R4+,R1L
          MOV.B       R1L,@PCR2     ;SCL = "0", SDA = "Dn Output"
          RTS
;
RD_data   .equ        $
          MOV.B       R0L,@PDR2     ;Input read data bitn
          MOV.B       R2L,@PCR2     ;SCL = "0", SDA = "Dn input"
          MOV.B       R0L,@PDR2
          MOV.B       R0L,@PCR2     ;SCL = "1", SDA = "Dn input"
          MOV.B       @PDR2,R6L
          MOV.B       R6L,@R4       ;Store serial code at Internal RAM
          ADDS        #1,R4
          MOV.B       R0L,@PDR2
          MOV.B       R2L,@PCR2     ;SCL = "0", SDA = "Dn input"
          RTS
;
PARCODE   .equ         $
          MOV.W       #SERAREA,R4   ;Load serial code address
          MOV.B       #SERSIZE,R5L
ANDDATA   MOV.B       @R4,R0L       ;AND.B #1,@SERAREA(H'FB80-FEBF)
          AND         #H'01,R0L
          MOV.B       R0L,@R4
          ADDS        #1,R4
          DEC         R5L
          BNE         ANDDATA
          MOV.W       #SERAREA,R4   ;Load serial code address
          MOV.W       #SPLSIZE,R5
          MOV.W       #SPLPGM,R6    ;Load execution sample program address
SERPAR    MOV.B       @R4+,R0L      ;Load serial code bit0
          MOV.B       @R4+,R1L      ;Load serial code bit1
          SHLL        R1L           ;Convert bit1 code
          ADD.B       R1L,R0L       ;Calculate Bit1,0 code
          MOV.B       @R4+,R1L      ;Load serial code bit2
          SHLL        R1L           ;Convert bit2 code
          SHLL        R1L
          ADD.B       R1L,R0L       ;Calculate Bit2-0 code
          MOV.B       @R4+,R1L      ;Load serial code bit3
          SHLL        R1L           ;Convert bit3 code
          SHLL        R1L
```

```
          SHLL          R1L
          ADD.B         R1L,R0L       ;Calculate Bit3-0 code
          MOV.B         @R4+,R1L      ;Load serial code bit4
          SHLL          R1L           ;Convert bit4 code
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          ADD.B         R1L,R0L       ;Calculate Bit4-0 code
          MOV.B         @R4+,R1L      ;Load serial code bit5
          SHLL          R1L           ;Convert bit5 code
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          ADD.B         R1L,R0L       ;Calculate Bit5-0 code
          MOV.B         @R4+,R1L      ;Load serial code bit6
          SHLL          R1L           ;Convert bit6 code
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          ADD.B         R1L,R0L       ;Calculate Bit6-0 code
          MOV.B         @R4+,R1L      ;Load serial code bit7
          SHLL          R1L           ;Convert bit7 code
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          SHLL          R1L
          ADD.B         R1L,R0L       ;Calculate Bit7-0 code
          MOV.B         R0L,@R6
          ADDS          #1,R6
          DEC           R5L
          BNE           SERPAR
          RTS
;*************************************************************************************
;*  H8/3644 Sample program of LED Control                                            *
;*************************************************************************************
;
          .org          H'2000
SAMPLE    BSET          #3,@H'FFEA    ;Initialize P73 Output Port
LEDCTL    BSET          #3,@H'FFDA    ;Turn on LED
          BSR           WAIT
          BCLR          #3,@H'FFDA    ;Turn off LED
          BSR           WAIT
          BRA           LEDCTL
WAIT      SUB.W         R0,R1
          MULXU         R0L,R2
          BNE           WAIT
          RTS
;
          .end
```

Revision Record

		Description	
Rev.	Date	Page	Summary
1.00	Dec.19.03	—	First edition issued

Keep safety first in your circuit designs!

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

Notes regarding these materials

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