

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.

H8/300H Tiny Series

Example of Connecting the MAXIM $\Sigma\Delta$ A/D Conversion (16 Bits)

Introduction

The 16-bit MAX1408 $\Sigma\Delta$ A/D conversion results are serially input to the H8/36014 CPU by providing a clock, and the outputs are displayed on an array of 7-segment LEDs as a 4-digit hexadecimal number.

Target Device

H8/300H Tiny Series H8/36014 CPU

Contents

1.	Specifications	2
2.	Description of Functions	6
3.	Description of Operation	9
4.	Description of Software	11
5.	Flowchart.....	17
6.	Program Listing	27

1. Specifications

- Figure 1 shows an example of the hardware configuration required to connect a serial output analog/digital converter (hereinafter referred to as an ADC). Analog voltages are input to analog input pin 0 (IN0 pin) of the ADC and are then A/D converted by the 16-bit delta sigma modulator.
- The conversion results are input to the microcomputer by using the clock applied by the microcomputer through synchronous serial communication.
- The received 16-bit data is converted to a 4-digit hexadecimal number and the result is displayed on a 7-segment LED array.

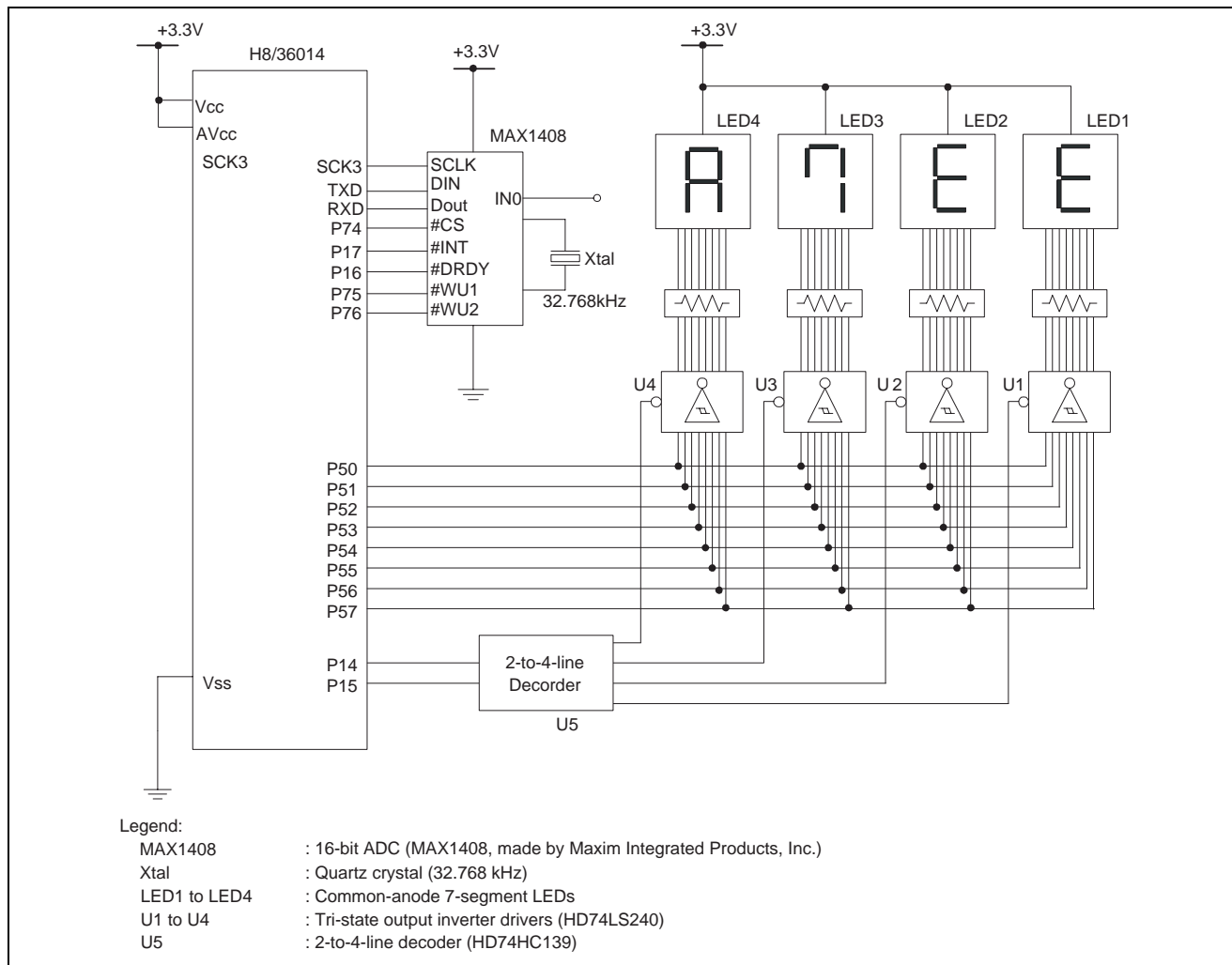


Figure 1 Hardware Configuration

- In this sample task, the operating voltage (Vcc) and analog power supply voltage (AVcc) of the H8/36014 are both 3.3 V and the OSC clock frequency is 10 MHz.
- The ADC used in this sample task is a serial-output analog digital converter (model MAX1408) manufactured by Maxim Integrated Products, Inc. The specifications are as follows:
 - Characteristics of the MAX1408.
 - Power voltage: +2.7 V to +3.6 V
 - Power consumption: 1.15mA (2.5 μ A in the sleep mode)

- c. Package: 28-pin SSOP
- d. Multi-channel 16-bit sigma-delta ADC
- B. The MAX1418 incorporates a delta-sigma modulator for A/D conversion. Since the converter has a high 16-bit resolution, it can be used for medical equipment, industrial controllers, portable devices, automatic metering, and robots.
- 6. The circuit in this sample task operates as follows.
 - A. Voltages input from analog pin 0 (IN0) of the MAX1418 are delta-sigma converted in cooperation with an external crystal resonator.
 - B. The 16-bit A/D converted data is input to the microcomputer by providing a serial clock.
 - C. The input 16-bit data is converted to a 4-digit hexadecimal number and displayed on the LED array.
 - D. For example, when a voltage of 0.82 V is applied to the IN0 pin, A/D conversion produces "1010 0111 1110 1110". The microcomputer then latches and converts it and then generates hexadecimal "A7EE" for the LED display.
 - E. A reference voltage of "1.25" is displayed on the LEDs as "FFFF" while 0 V is displayed as "0000".
- 7. In this sample task, the 7-segment LED display is set up by connecting the port outputs to the tri-state output inverter drivers (HD74LS240) and connecting the driver outputs to the cathodes of the 7-segment LEDs. In addition, all the ports used for the four 7-segment LEDs are connected to the 7-segment LEDs and the enable pins of the tri-state inverter drivers are used to renew between the 7-segment LEDs. The signal generation for renewing between the LEDs is controlled by the two port outputs of a 2-to-4 line decoder (HD74HC139). Figure 2 illustrates the method of controlling the 7-segment LEDs.

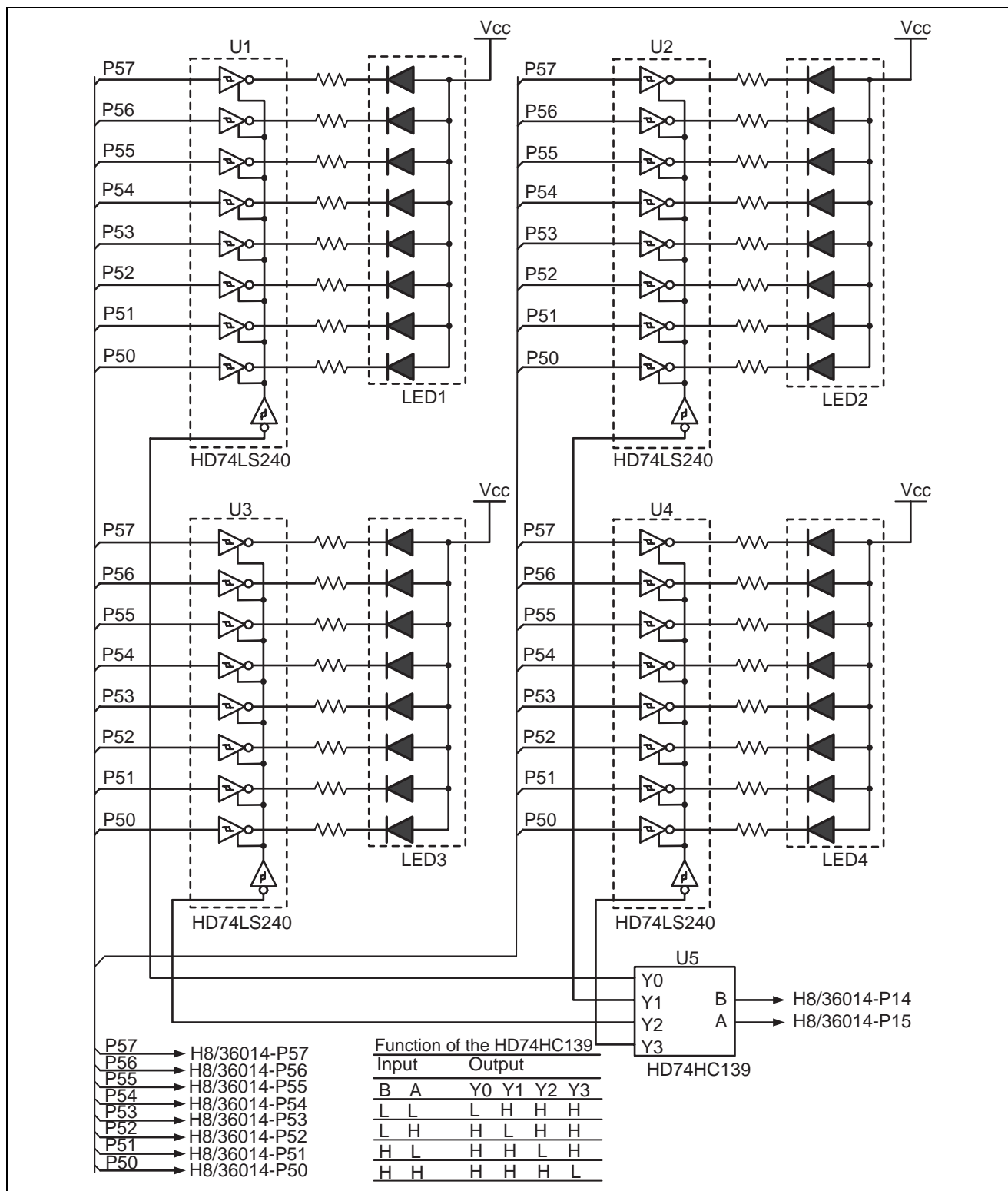


Figure 2 Method of Controlling 7-Segment LEDs

8. In this sample task, the 16-bit data obtained as the result of conversion by the MAX1408 $\Sigma\Delta$ A/D, is displayed on the 7-segment LED array as a 4-digit hexadecimal number. Figure 3 shows how the result of MAX1408 $\Sigma\Delta$ A/D conversion is displayed on the LEDs.

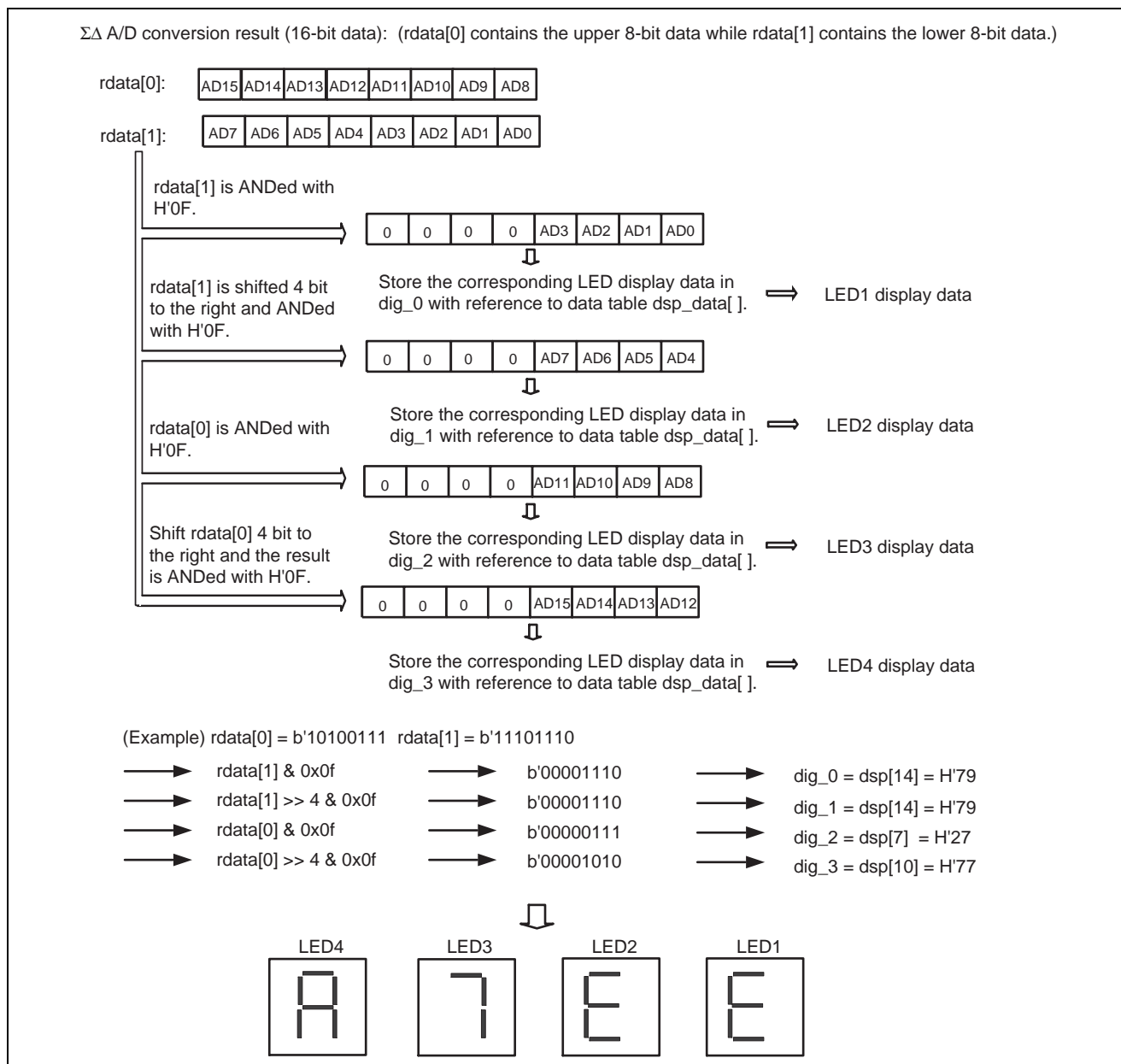


Figure 3 Display of Result of MAX1408 $\Sigma\Delta$ A/D Conversion on the LEDs

2. Description of Functions

1. Figure 4 is a block diagram of the H8/36014 functions used in this task. Table 1 lists the function allocations.

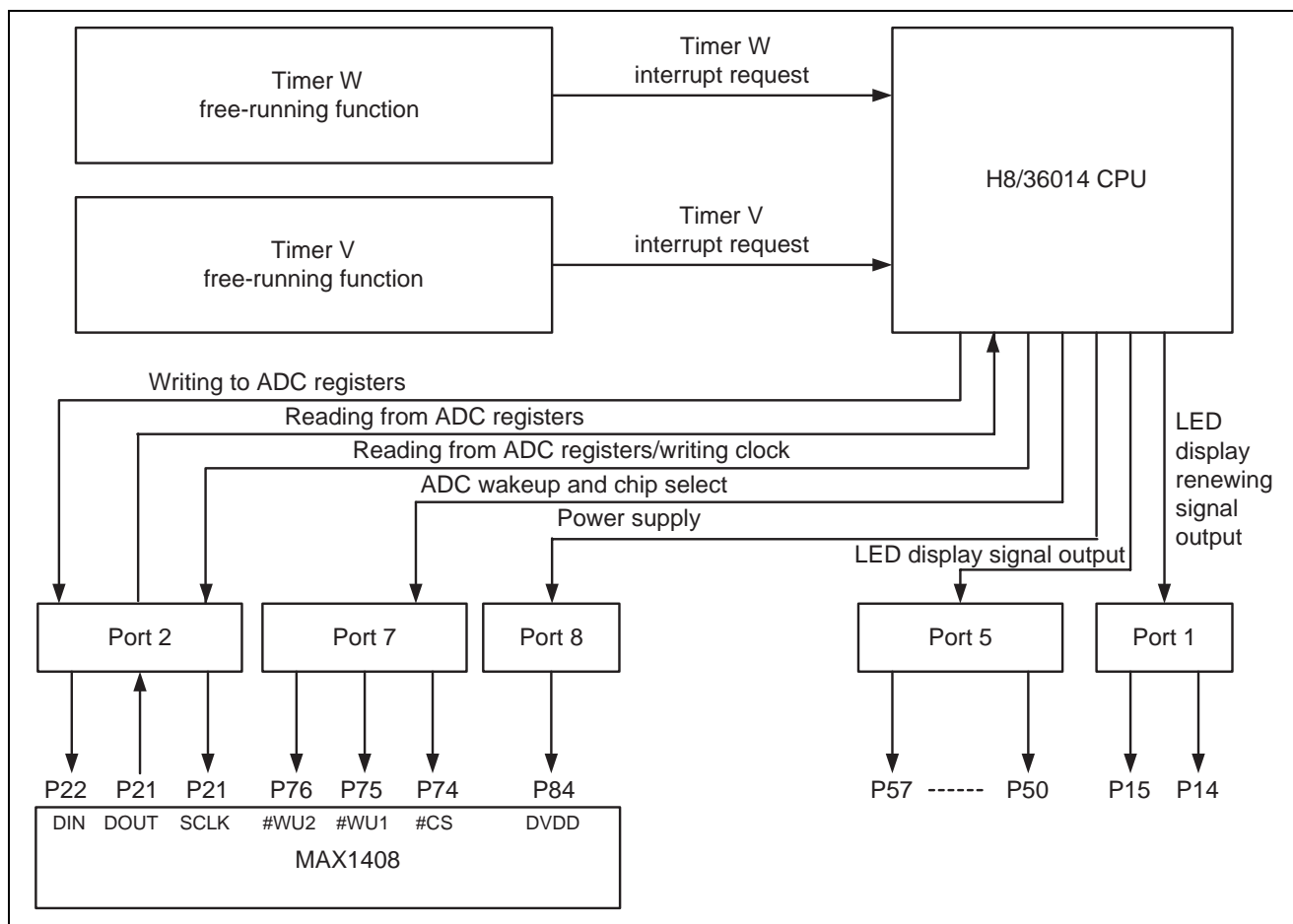


Figure 4 Block Diagram of Functions Used

Table 1 Function Assignments

Function	Function assignment
Timer W	The free-running function of timer W is used to convert the DATA register value, obtained as a result of MAX1408 $\Sigma\Delta$ A/D conversion, to LED display data. Then it is stored into RAM.
Timer V	The free-running function of timer V is used to control the renewing the display between the 7-segment LEDs. Each of the four 7-segment LEDs is lit in sequence at intervals of 3.2768 ms (the time at which timer V overflows), enabling dynamic illumination of the LEDs.
Port 1	The P14 and P15 output pins of port 1 are used to renew the display on the four 7-segment LEDs. The P14 and P15 output pins are connected to the I/O pins of a 2-to-4-line decoder.
Port 2	The P22 output pin of port 2 is used to write to the MAX1408 $\Sigma\Delta$ ADC registers. The P20 output pin of port 2 is used to set the clock for writing to or reading from the MAX1408 $\Sigma\Delta$ ADC registers. The P21 input pin of port 2 is used to read from the MAX1408 $\Sigma\Delta$ ADC registers.
Port 5	The P50-P57 output pins of port 5 are used to display data on the currently active 7-segment LED. The 10-bit data, obtained as a result of the MAX1408 $\Sigma\Delta$ A/D conversion, is converted to 4-digit hexadecimal display data and output to the LED.
Port 7	The P74 to P76 output pins of port 7 are used to wake up MAX1408 $\Sigma\Delta$ ADC and to select chips.
Port 8	The P84 output pin is used to connect to the power supply of the MAX1408 $\Sigma\Delta$ ADC.

2. Figure 5 shows how the 7-segment LED used in this task is connected. A high output from port 5 lights the corresponding segment of the LED, as shown in the figure. Table 2 lists the relationships between port 5 outputs and display on the LED display.

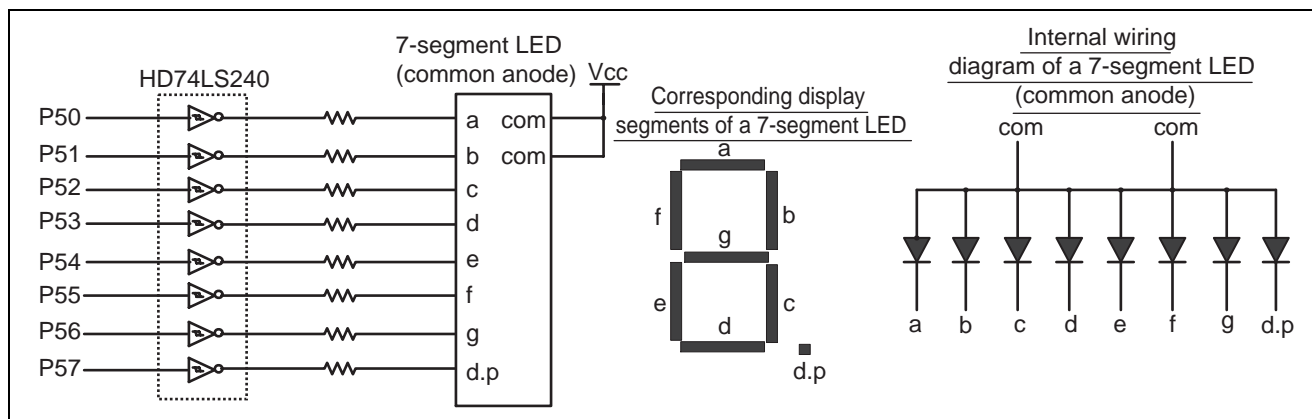


















Figure 5 Connection Diagram and Internal Connections of 7-Segment LED

Table 2 Relationship between Port 5 Outputs and 7-Segment LED Display Data

LED display	Port 5 output data								LED display	Port 5 output data							
	P57	P56	P55	P54	P53	P52	P51	P50		P57	P56	P55	P54	P53	P52	P51	P50
	0	0	1	1	1	1	1	1		0	1	1	1	0	1	1	1
	0	0	0	0	0	1	1	0		0	1	1	1	1	1	0	0
	0	1	0	1	1	0	1	1		0	0	1	1	1	0	0	1
	0	1	0	0	1	1	1	1		0	1	0	1	1	1	1	0
	0	1	1	0	0	1	1	0		0	1	1	1	1	0	0	1
	0	1	1	0	1	1	0	1		0	1	1	1	0	0	0	1
	0	1	1	1	1	1	0	1									
	0	0	1	0	0	1	1	1									
	0	1	1	1	1	1	1	1									
	0	1	1	0	1	1	1	1									

3. Description of Operation

- Figure 6 shows how the 16-bit DATA register value, obtained as a result of MAX1408 $\Sigma\Delta$ A/D conversion, is converted to LED display data using timer W and then stored into RAM. In this sample task, the result of MAX1408 $\Sigma\Delta$ A/D conversion is converted to LED display data and stored into RAM at the timer W overflow flag in the tmrw routine, as shown in this figure.

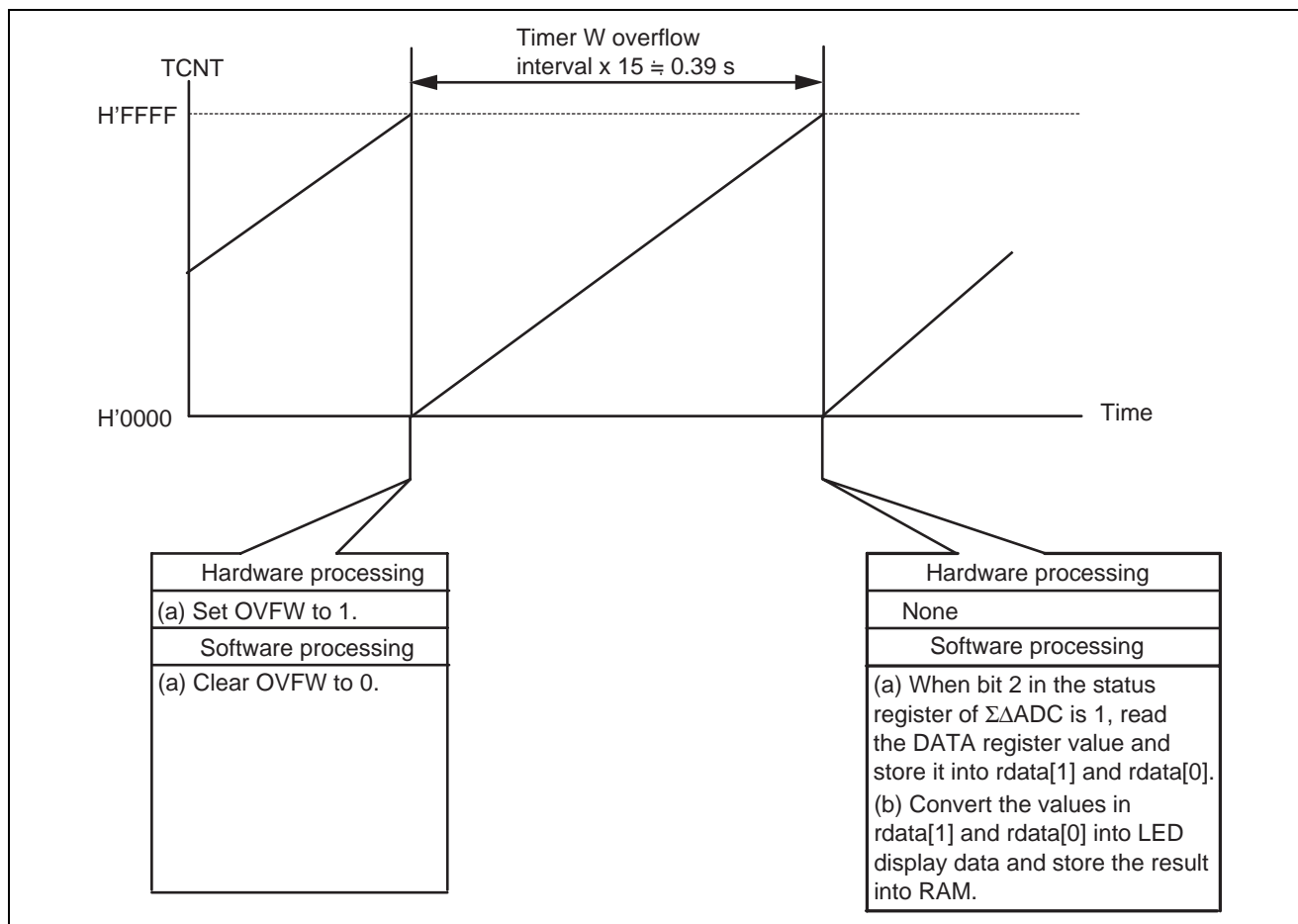


Figure 6 Storing the $\Sigma\Delta$ A/D Conversion Result into RAM with Timer W

2. The following describes the descriptions of 7-segment LED operation. Figure 7 shows how a value of "A7EE" is displayed on LED4 to LED1. As shown in the figure, each of LED1 to LED4 is lit in sequence at the timer V overflow interval, resulting in dynamic display on the 7-segment LED.

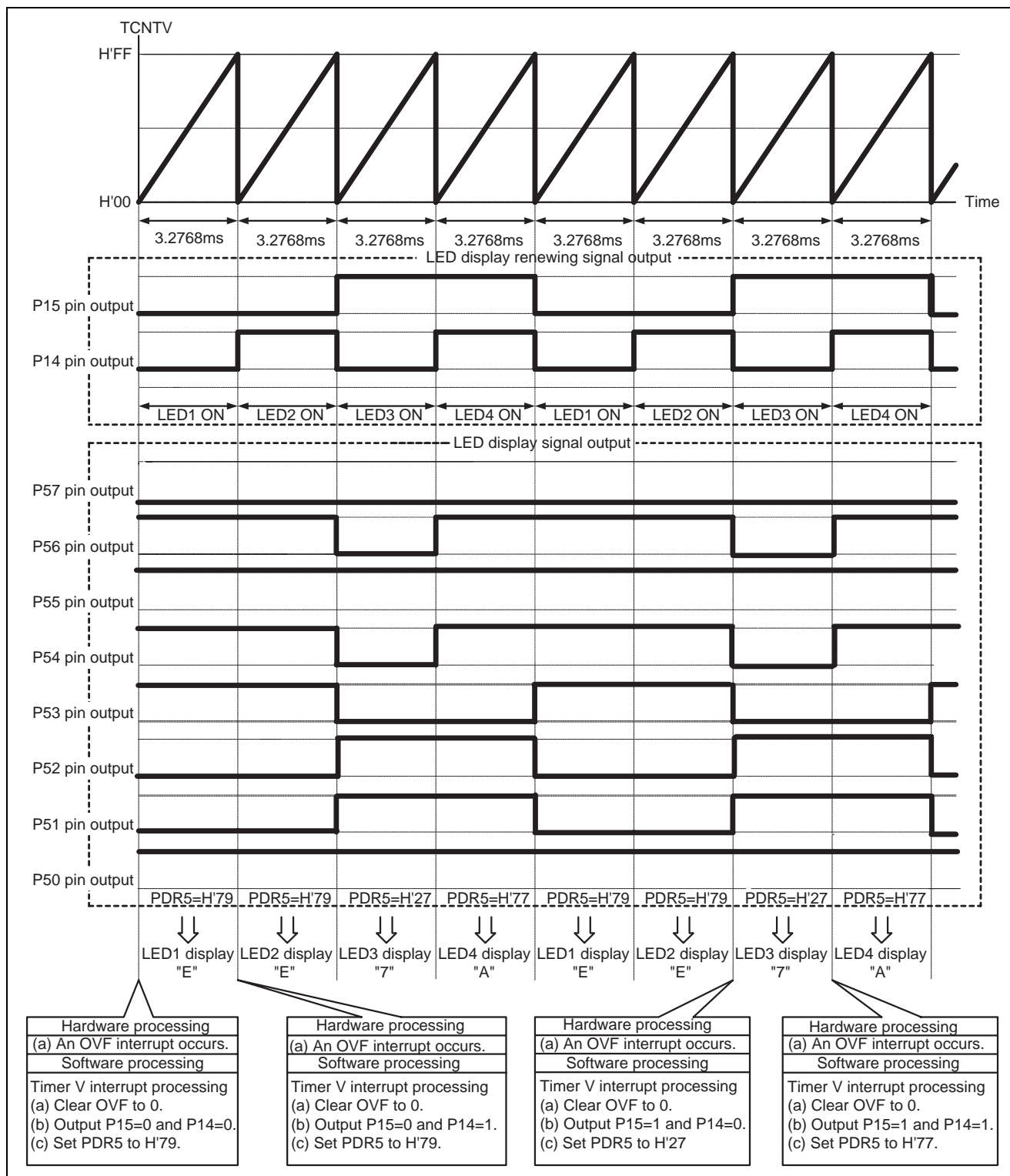


Figure 7 Descriptions of 7-Segment LED Display Control

4. Description of Software

1. Modules

Table 3 lists the modules used in this sample task.

Table 3 Modules

Module name	Label name	Function
Main routine	main	Makes the initial settings, calls the initial setting function for the $\Sigma\Delta$ A/D converter, and then enables an interrupt.
Timer W interrupt processing routine	tmrw	Clears the interrupt flag. This routine also converts the DATA register value, obtained as a result of MAX1408 $\Sigma\Delta$ A/D conversion, to LED display data after bit 2 of the STATUS register is set to 1, and then stores it into RAM.
Timer V interrupt processing routine	tmrv	Clears the interrupt flag, outputs the LED display data, and controls LED display switching.
ADC_Init() processing routine	ADC_Init	Initializes the MAX1408 $\Sigma\Delta$ A/D converter.
DataIn() processing routine	DataIn	Writes to or reads from the MAX1408 $\Sigma\Delta$ A/D converter registers.
DataOut() processing routine	DataOut	Writes to the MAX1408 $\Sigma\Delta$ A/D converter registers.

2. Arguments

No arguments are used in this sample task.

3. Internal Registers

Table 4 lists the internal registers used in this task.

Table 4 Internal Registers

Register name	Description	Address	Setting
TCRV0	Timer control register V0: Selects an input clock of TCNTV, specifies the clear conditions and controls each interrupt request.	H'FFA0	H'03 (At initial setting)
CMIEB	Compare match interrupt enable B: Prohibits an interrupt request by CMFB of TCSRv when the bit is 0.	Bit 7	0
CMIEA	Compare match interrupt enable A: Prohibits an interrupt request by CMFA of TCSRv when the bit is 0.	Bit 6	0
OVIE	Timer overflow interrupt enable: Prohibits an interrupt request by OVF of TCSRv when the bit is 0. Enables an interrupt request by OVF of TCSRv when the bit is 1.	Bit 5	0/1
CCLR1	Counter clear 1 to 0:	Bit 4	0
CCLR0	Specifies the clear conditions of TCNTV. When CCLR1=0 and CCLR0=0 are set, clearing TCNTV is prohibited.	Bit 3	0

Register name	Description	Address	Setting
CKS2	Clock select 2 to 0:	Bit 2	0
CKS1	Selects the clock and count conditions to input to TCNTV in combination with TCRV1 and ICKS0.	Bit 1	1
CKS0	When CKS2 = 0, CKS1 = 1, CKS0 = 1, and ICKS0 = 1 are set, TCNTV performs a count at the falling edge whose internal clock is $\phi/128$.	Bit 0	1
TCSR	Timer control/status register V:	H'FFA1	H'10
CMFB	Compare match flag B: Sets to 1 when the values of TCNTV and TCORB match.	Bit 7	0
CMFA	Compare match flag A: Sets to 1 when the values of TCNTV and TCORA match.	Bit 6	0
OVF	Timer overflow flag: Sets to 1 when the value of TCNTV overflows. Clears to 0 when a zero is written to OVF after reading OVF in the OVF=1 status.	Bit 5	0
OS3	Output select 3 to 2: Sets the output levels of the TMOV pins with compare match B.	Bit 3	0
OS2	When OS3=0 and OS2=0 are sets, nothing changes.	Bit 2	0
OS1	Output select 1 to 0: Sets the output levels of the TMOV pin with compare match A.	Bit 1	0
OS0	When OS1 = 0 and OS0 = 0 are set, nothing changes.	Bit 0	0
TCRV1	Timer control register V1:	H'FFA5	H'E2
TVEG1	TRGV input edge select 1 to 0:	Bit 4	0
TVEG0	Selects the input edge of the TRGV pins When TREG1 = 0 and TREG0 = 0 are set, the trigger input from the TRGV pins is prohibited.	Bit 3	0
TRGE	TRGV input enable: Enables or prohibits the TCNTV count-up by the edge input selected in TVEG1 and TVEG0. When TREG = 0 is set, the startup of the TCNTV count-up by the pin input of TRGV and the stop of TCNTV count-up are prohibited if TCNTV is cleared by a compare match.	Bit 2	0

Register name	Description	Address	Setting
ICKS0	Internal clock select 0: Selects the clock and count conditions to input to TCNTV depending on the combination of CKS2 to CKS0 of TCRV0. When CKS2 = 0, CKS1 = 1, CKS0 = 1, and ICKS0 = 1 are set, TCNTV performs a count at the falling edge whose internal clock is $\phi/128$.	Bit 0	1
TMRW	Timer mode register W: Selects the general register function and the output mode.	H'FF80	H'80
CTS	Count startup When CTS = 1, TCNT indicates the startup of the counter. When CTS = 0, TCNT indicates the stop of the counter.	Bit 7	1
TCRW	Timer control register W: Selects the counter clock. Sets the clear conditions of a counter and the output level of a counter.	H'FF81	H'30
CKS2	Clock select:	Bit 6	0
CKS1	When CKS2 = 0, CKS1 = 1, and CKS0 = 0,	Bit 5	1
CKS0	this function sets the TCNT input clock to the 4-frequency divided clock of the system clock.	Bit 4	0
TIEWR	Timer interrupt enable register W: Controls the interrupt request to timer W.	H'FF82	H'00 (At initial setting)
OVIE	Timer overflow interrupt enable: Prohibits an interrupt request by OVF when OVIE = 0. Enables an interrupt request by OVF when OVIE = 1	Bit 7	0/1
TSRW	Indicates the interrupt request status.	H'FF83	H'00
OVF	Timer overflow: Indicates that TCNT does not overflow when OVF = 1. Indicates that TCNT overflows when OVF = 1.	Bit 7	0
TCNT	Timer counter: This is a 16-bit up-counter used as the input of the 8-frequencydivided clocks of the system clock.	H'FF86	H'00
PMR1	Port mode register 1: Sets the pin function for ports 1, 2, and 7.	H'FFE0	H'00
IRQ3	Switching the pin function of P17/_IRQ3/TRGV: Provides the function of the general I/O port of P17 when the bit is 0.	Bit 7	0
IRQ0	Switching the pin function of P14/_IRQ0: Provides the function of the general I/O port of P14 when the bit is 0.	Bit 4	0

Register name	Description	Address	Setting
TXD2	Switching the pin function of P72/TXD_2: Provides the function of the general I/O port of P72 when the bit is 0.	Bit 3	0
TXD	Switching the pin function of P22/TXD	Bit 1	0
PCR1	Port control register 1: Selects the I/O of the pin used as a general I/O port of port 1 for each bit. When PCR1=H'38, P17 and P16, and P12 to P10 function as a general input pin. P14 and P15 function as a general output pin.	H'FFE4	H'38
PDR1	Port data register 1: This is a general I/O port data register of port 1.	H'FFD4	H'08
PUCR1	Port pull-up control register 1: Controls the pull-up MOS of each pin of port 1 set to the input port for the bit. When PUCR1 = H'08, the pull-up MOS for P17 to P14 and P12 to P10 are set to off.	H'FFD0	H'08
PDR2	Port data register 2 This is a general I/O port data register of port 2.	H'FFD5	H'F8
PCR2	Port control register 2: Selects the I/O of the pin used as a general I/O port of port 2 for the bit. When PCR2 = H'FD, P21 functions as a general input pin. P20 and P22 function as a general output terminal.	H'FFE5	H'FD
PMR5	Port mode register 5: Sets the pin function of port 5.	H'FFE1	H'00
POF7	Switching the function of P57: Provides the general I/O port function of P57 when the bit is 0.	Bit 7	0
POF6	Switching the function of P56: Provides the general I/O port function of P56 when the bit is 0.	Bit 6	0
WKP5	Switching the pin function of P55/_WKP5/_ADTRG: Provides the general I/O port function of P55 when the bit is 0.	Bit 5	0
WKP4	Switching the pin function of P54/_WKP4: Provides the general I/O port function of P54 when the bit is 0.	Bit 4	0
WKP3	Switching the pin function of P53/_WKP3: Provides the general I/O port function of P53 when the bit is 0.	Bit 3	0
WKP2	Switching the pin function of P52/_WKP2: Provides the general I/O port function of P52 when the bit is 0.	Bit 2	0

Register name	Description	Address	Setting
WKP1	Switching the pin function of P51/_WKP1: Provides the general I/O port function of P51 when the bit is 0.	Bit 1	0
WKP0	Switching the pin function of P50/_WKP0: Provides the general I/O port function of P50 when the bit is 0.	Bit 0	0
PUCR5	Port pull-up control register 5: Controls the pull-up MOS of port 5 set to the input port for the bit. When PUCR5 = H'00, the pull-up MOS of P57 to P50 is set to off.	H'FFD1	H'00
PDR5	Port data register 5: This is a general I/O port data register of port 5.	H'FFD8	H'00
PCR5	Port control register 5: Selects the I/O of the pins used as a general I/O port of port 5. When PCR5 = H'FF, P57 to P50 function as a general I/O pin.	H'FFE8	H'FF
PDR7	Port data register 7: This is a general I/O port data register of port 7.	H'FFDA	H'80
PCR7	Port control register 7: Selects the I/O of the pins used as a general I/O port of port 7 for the bit. When PCR7 = H'FF, P76 to P70 function as a general I/O pin.	H'FFEA	H'FF
PDR8	Port data register 8: This is a general I/O port data register of port 8.	H'FFD8	H'00
PCR8	Port control register 8: Selects the I/O of the pins used as a general I/O port of port 8. When PCR8 = H'FF, P84 to P80 function as a general I/O pin.	H'FFEB	H'FF

4. Description of RAM

Table 5 describes the RAM used in this sample task.

Table 5 Description of RAM

Label name	Description	Address	Module label name
dig_0	Stores LED1 display data (1 byte)	H'FB80	main, tmrw
dig_1	Stores LED2 display data (1 byte)	H'FB81	main, tmrw
dig_2	Stores LED3 display data (1 byte)	H'FB82	main, tmrw
dig_3	Stores LED4 display data (1 byte)	H'FB83	main, tmrw
cnt	8-bit counter for renewing display between LED1-LED4 (1 byte)	H'FB84	main, ttmrv

Label name	Description	Address	Module label name
counter_su b	8-bit counter to adjust an A/D acquisition interval (1 byte)	H'FB85	main, tmrw
sdata	Stores a register value for controlling MAX1408 $\Sigma\Delta$ A/D	H'FB87	tmrw, ADC_Init, DataIn, Data Out
rdata	Stores the DATA register value that results from MAX1408 $\Sigma\Delta$ A/D conversion.	H'FB8C	tmrw, DataIn,

5. Description of Data Tables

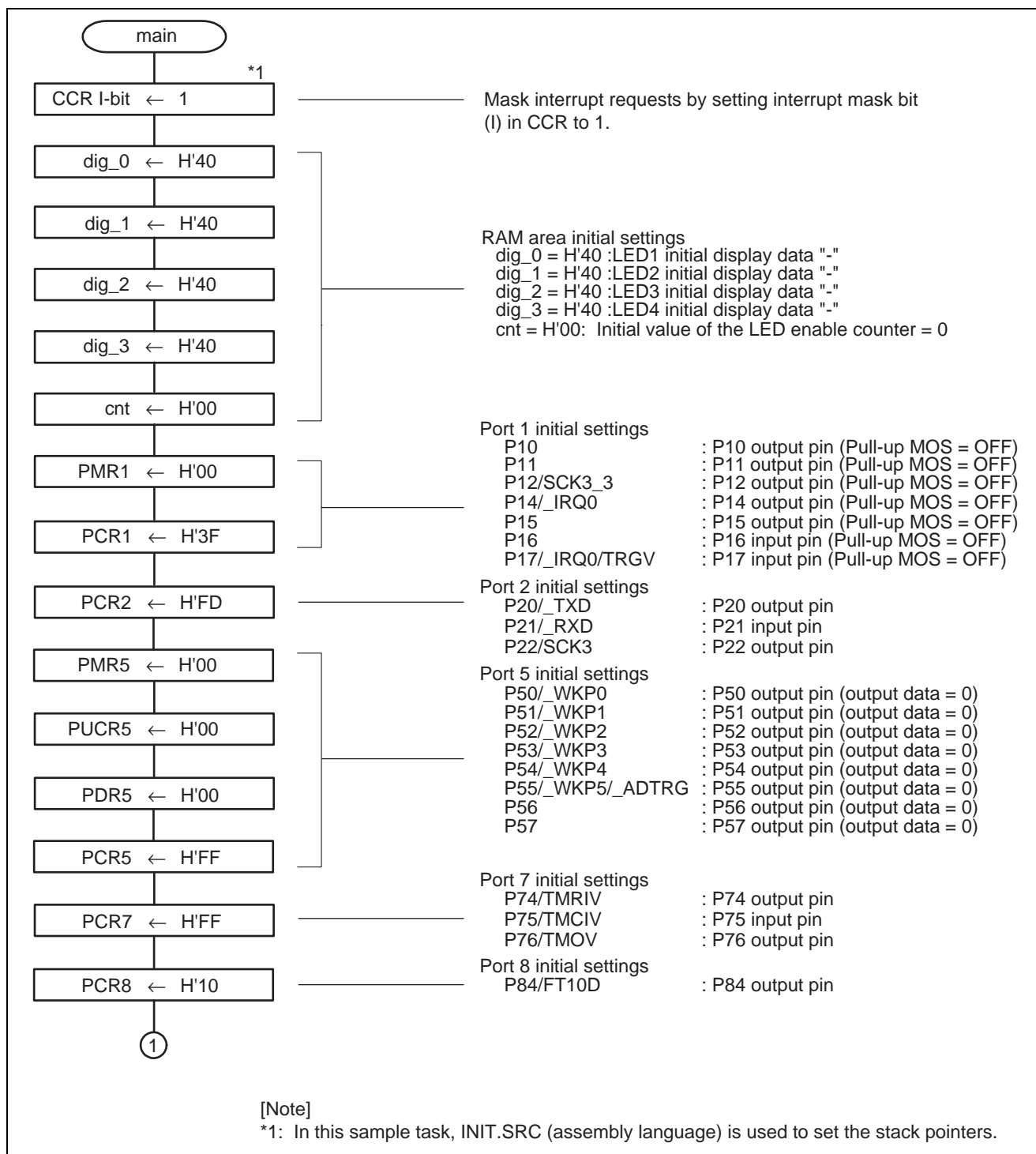
In this sample task, display data for the 7-segment LEDs is stored in ROM as a data table consisting of a 1-dimensional array (data table). Table 6 describes the data table for 7-segment LED display (dsp_data []).

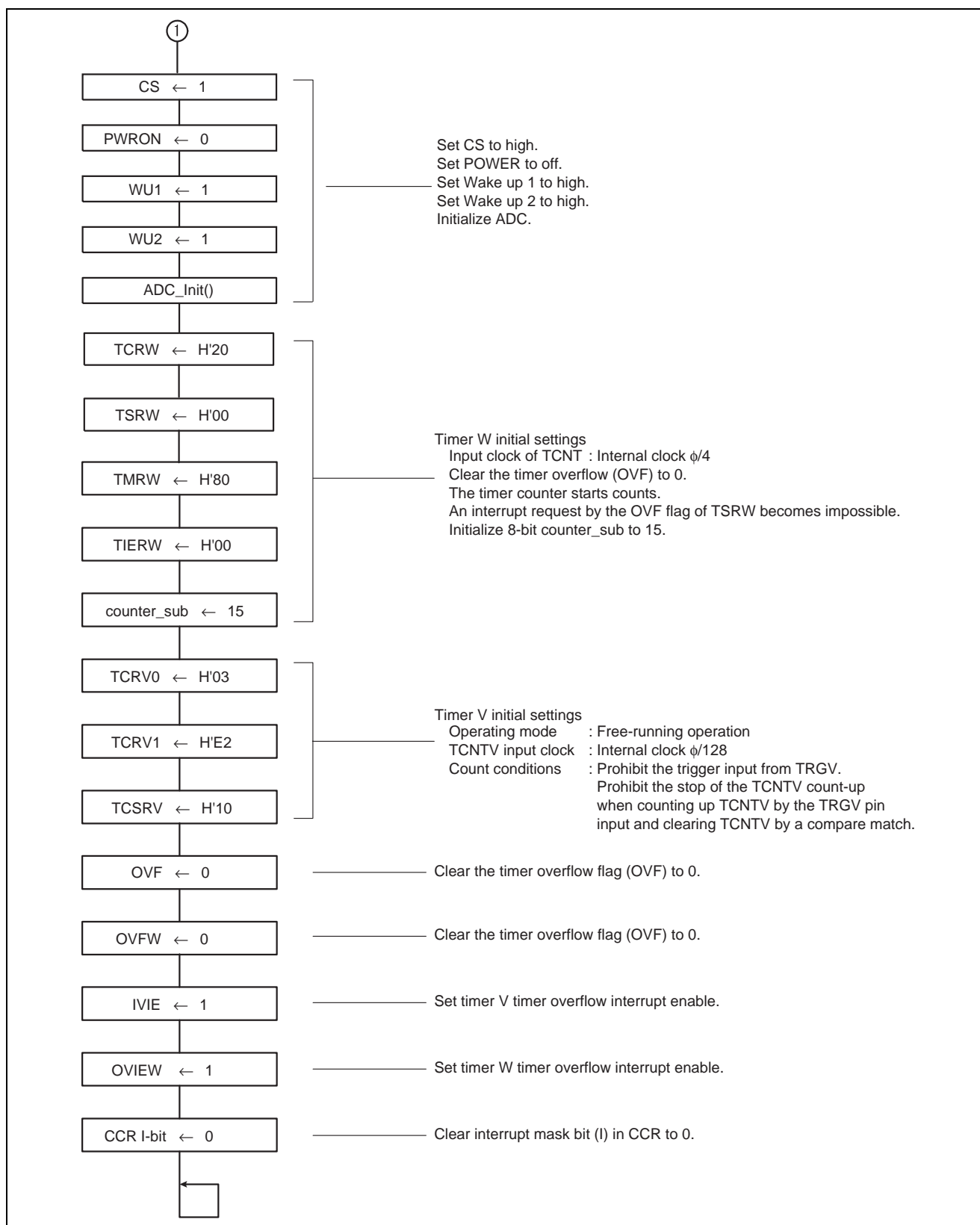
Table 6 Description of Data Table (dsp_data[]) for 7-Segment LED Display

Element	Data	Description	Data size	Address
dsp_data[0]	H'3F	Port 5 output data for displaying "0" on a LED	1 byte	H'72C
dsp_data[1]	H'06	Port 5 output data for displaying "1" on a LED	1 byte	H'72D
dsp_data[2]	H'5B	Port 5 output data for displaying "2" on a LED	1 byte	H'72E
dsp_data[3]	H'4F	Port 5 output data for displaying "3" on a LED	1 byte	H'72F
dsp_data[4]	H'66	Port 5 output data for displaying "4" on a LED	1 byte	H'730
dsp_data[5]	H'6D	Port 5 output data for displaying "5" on a LED	1 byte	H'731
dsp_data[6]	H'7D	Port 5 output data for displaying "6" on a LED	1 byte	H'732
dsp_data[7]	H'27	Port 5 output data for displaying "7" on a LED	1 byte	H'733
dsp_data[8]	H'7F	Port 5 output data for displaying "8" on a LED	1 byte	H'734
dsp_data[9]	H'6F	Port 5 output data for displaying "9" on a LED	1 byte	H'735
dsp_data[10]	H'77	Port 5 output data for displaying "A" on a LED	1 byte	H'736
dsp_data[11]	H'7C	Port 5 output data for displaying "B" on a LED	1 byte	H'737
dsp_data[12]	H'39	Port 5 output data for displaying "C" on a LED	1 byte	H'738
dsp_data[13]	H'5E	Port 5 output data for displaying "D" on a LED	1 byte	H'739
dsp_data[14]	H'79	Port 5 output data for displaying "E" on a LED	1 byte	H'73A
dsp_data[15]	H'71	Port 5 output data for displaying "F" on a LED	1 byte	H'73B

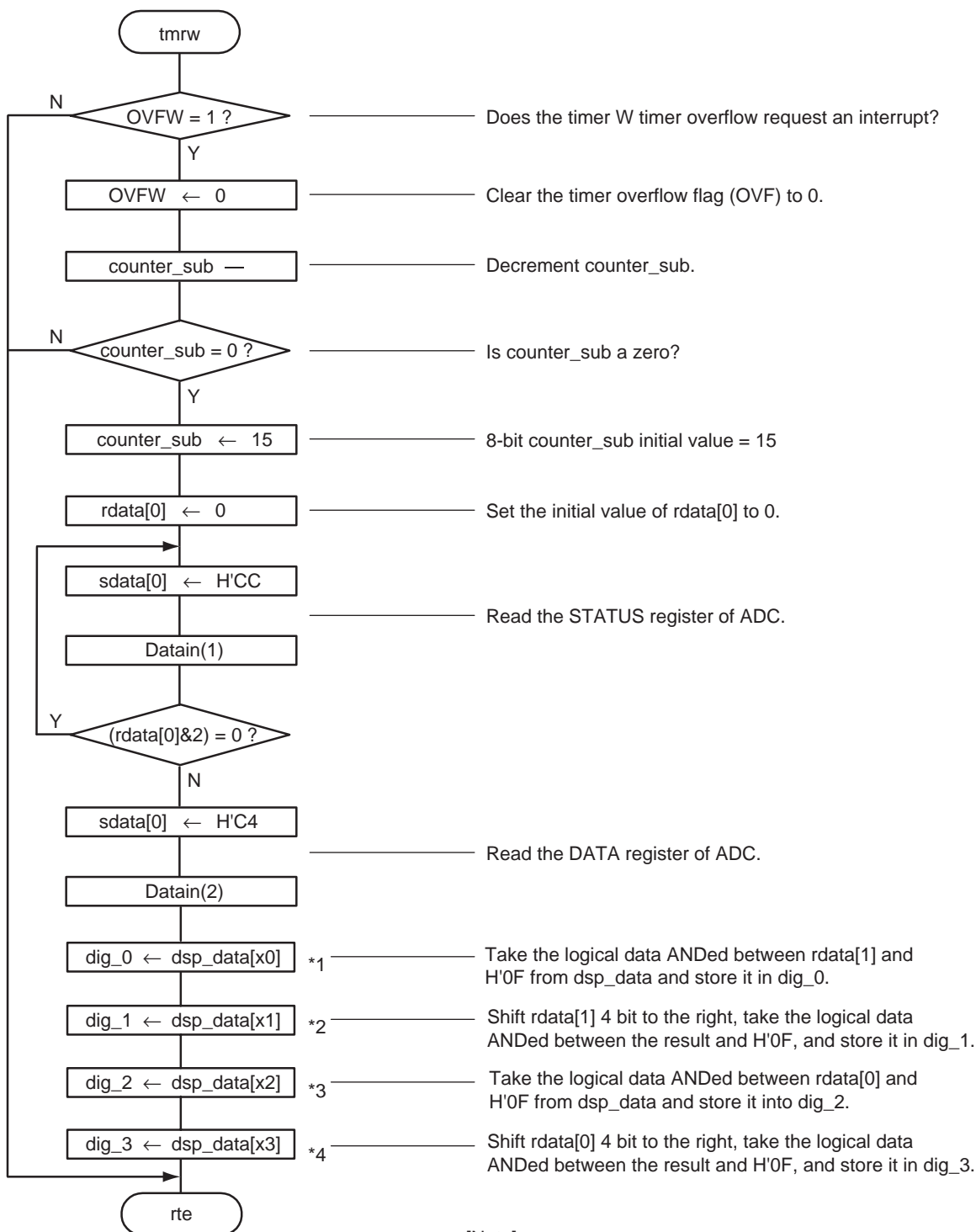
5. Flowchart

1. Main Routine (main)





2. Timer W Interrupt Processing Routine (tmrw)



[Note]

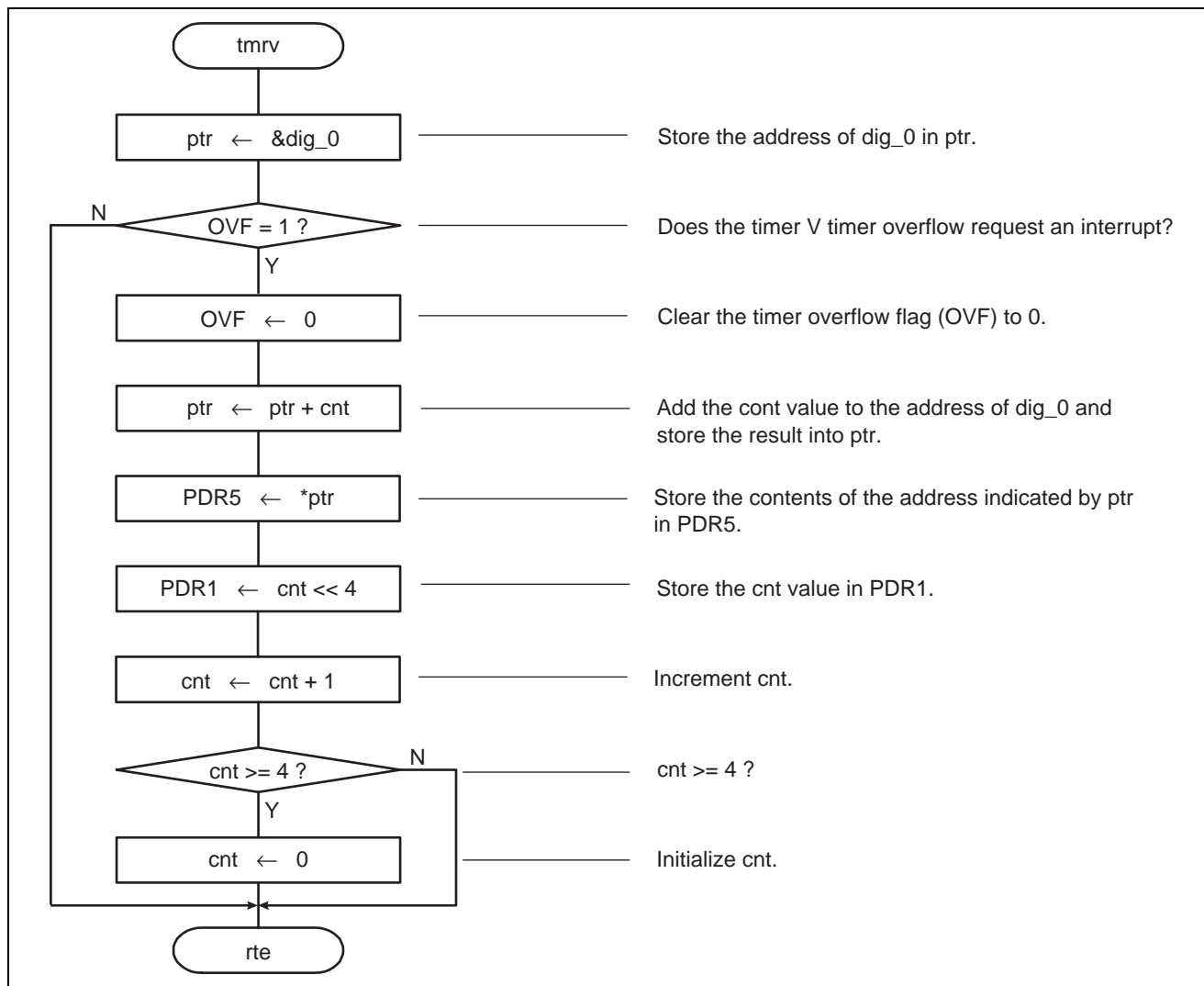
*1 : x0 = rdata[1] & H'0F

*2 : x1 = rdata[1] >> 4 & H'0F

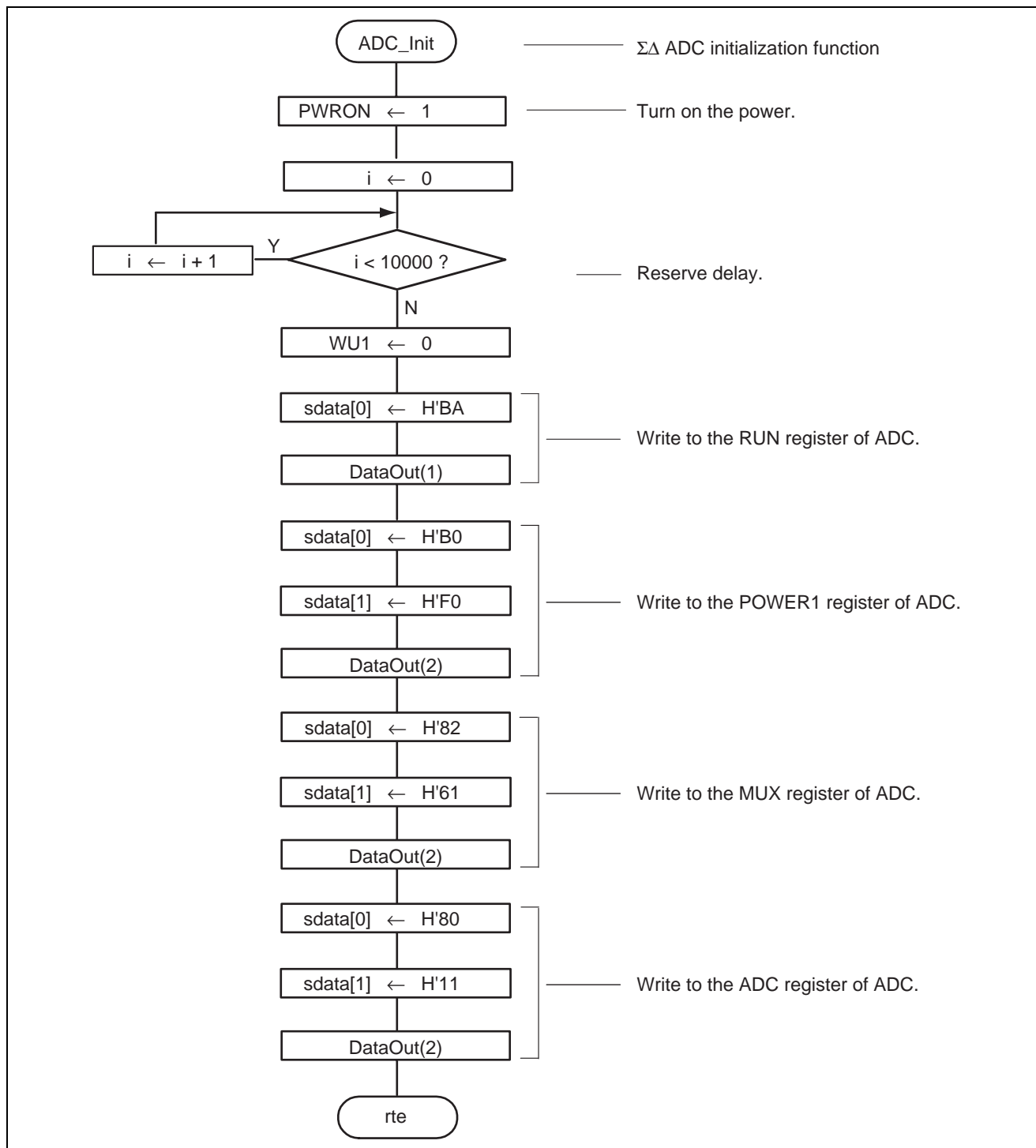
*3 : x2 = rdata[0] & H'0F

*4 : x3 = rdata[0] >> 4 & H'0F

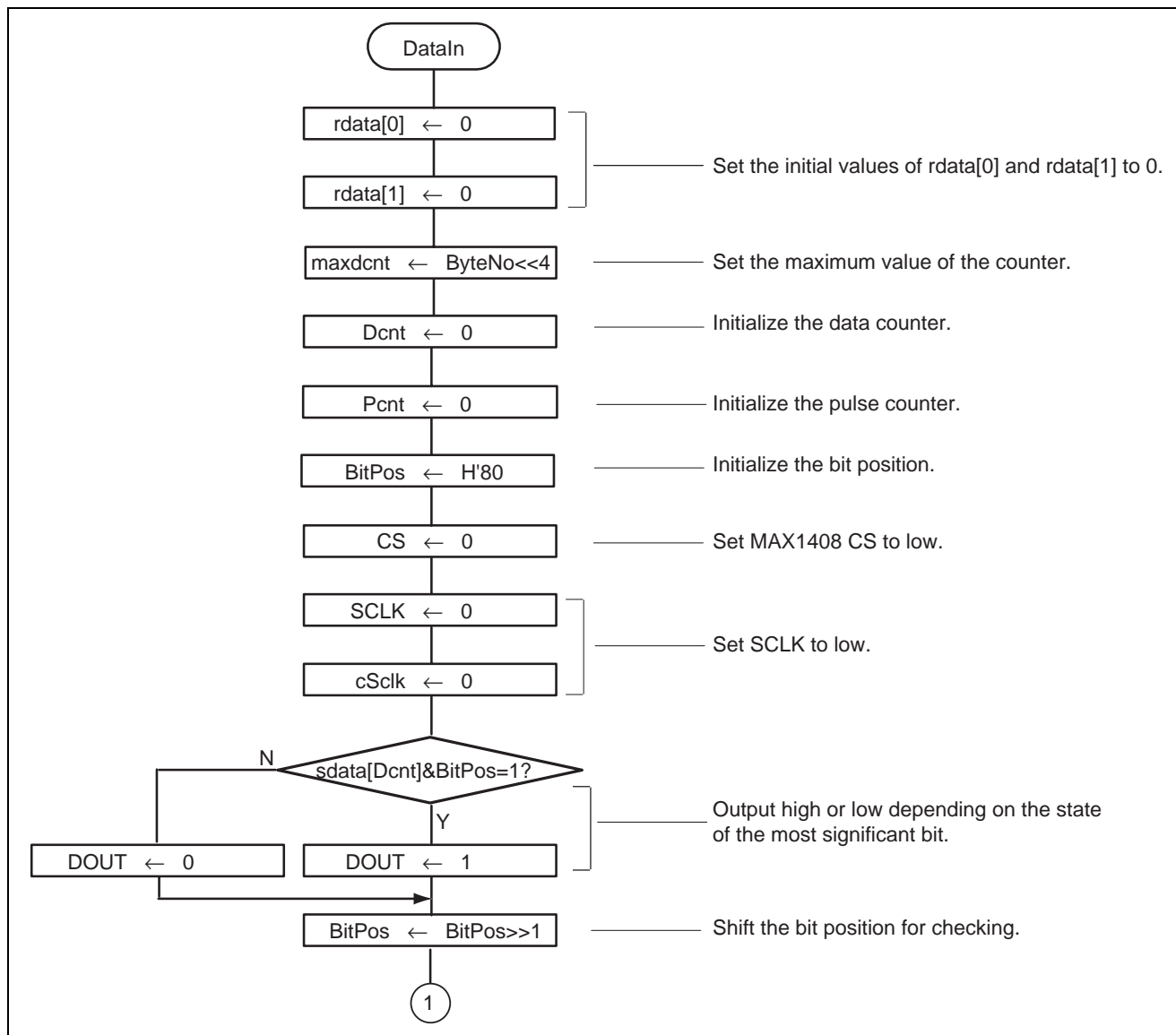
3. Timer V Interrupt Processing Routine (tmrv)

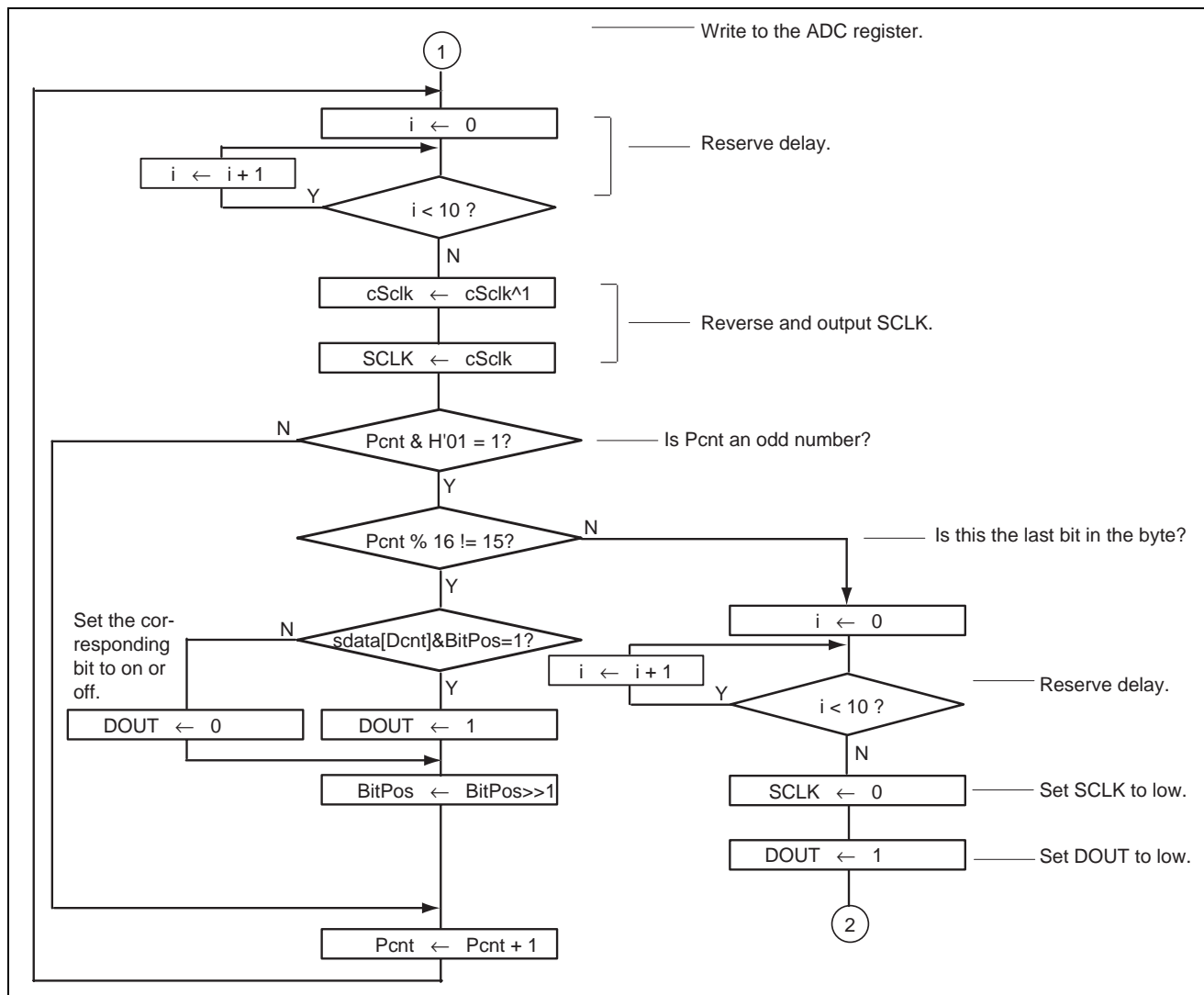


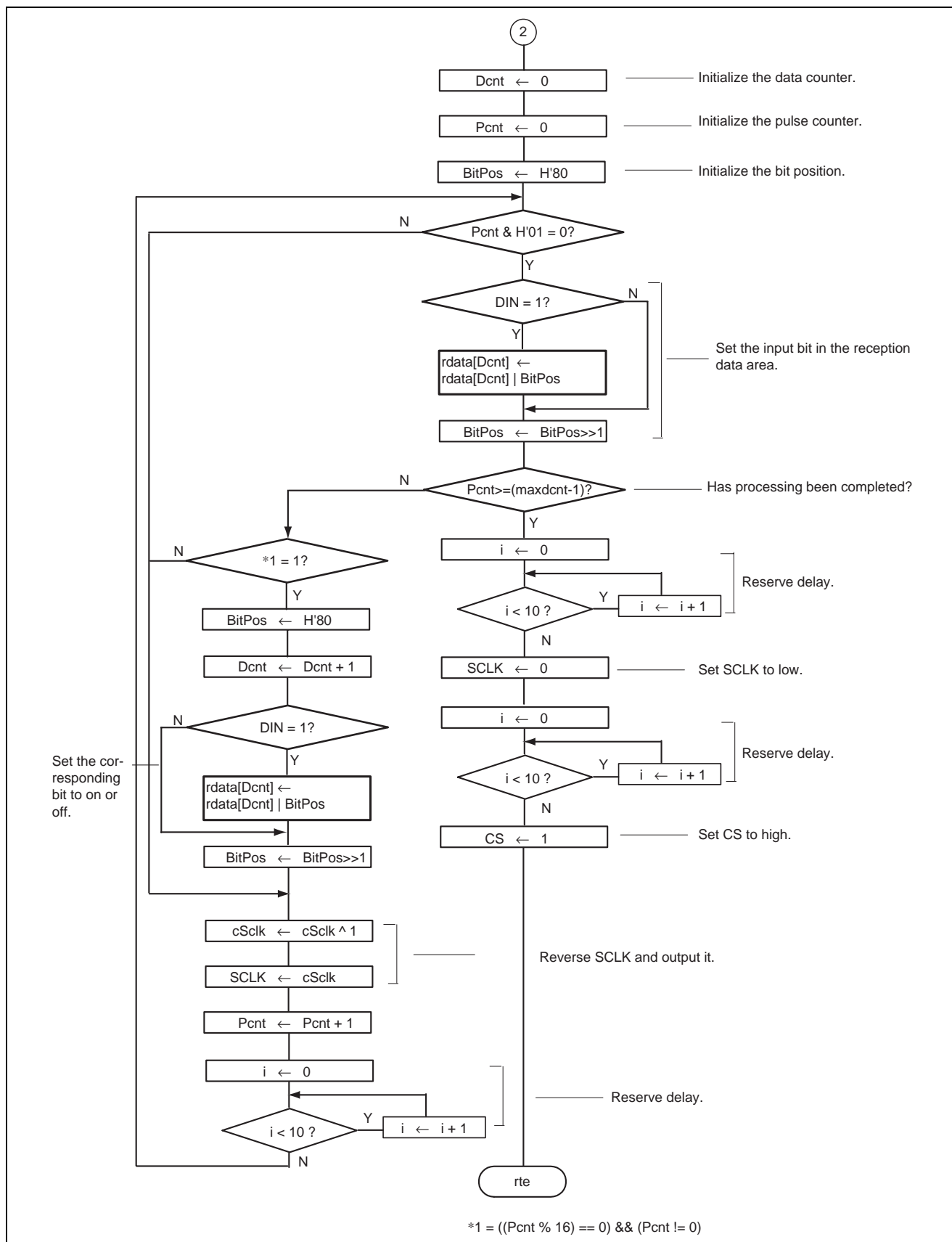
4. ADC_Init() Processing Routine



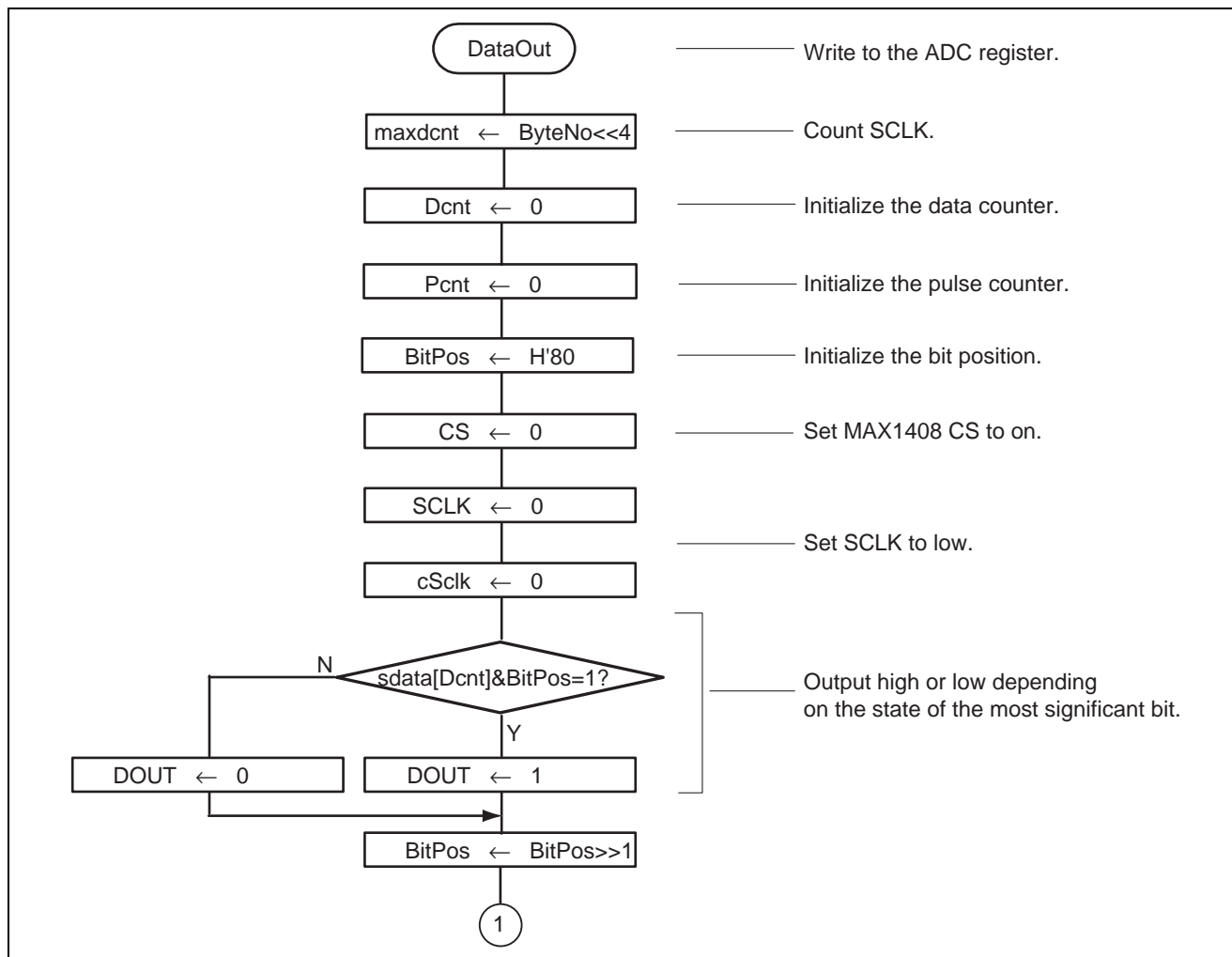
5. DataIn() Processing Routine

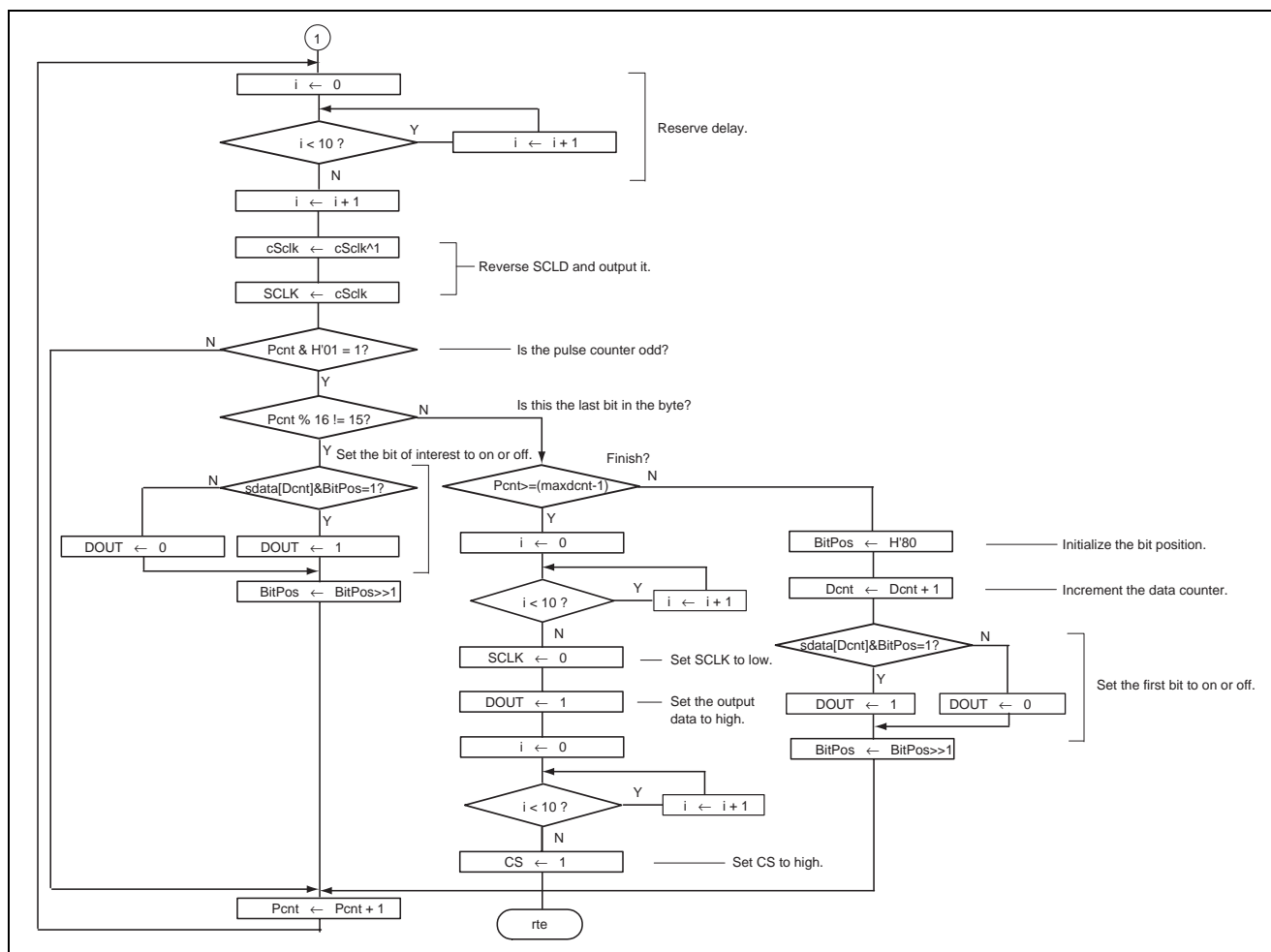






6. DataOut() Processing Routine





6. Program Listing

INIT.SRC(program list)

```
.export _INIT
.import _main

;
.section    P, CODE
_INIT:
    mov.w   #h'ff80, r7
    ldc.b   #b'10000000, ccr
    jmp @_main
;
.end
```

```
/* H8/300H tiny Series -H8/36014- Application note */
/* Application example */
/* Example of connecting to  $\Sigma\Delta$  A/D */
```

```
#include <machine.h>
```

```
/* Symbol definition */
```

```
struct BIT {
    unsigned char b7:1;    /* bit 7 */
    unsigned char b6:1;    /* bit 6 */
    unsigned char b5:1;    /* bit 5 */
    unsigned char b4:1;    /* bit 4 */
    unsigned char b3:1;    /* bit 3 */
    unsigned char b2:1;    /* bit 2 */
    unsigned char b1:1;    /* bit 1 */
    unsigned char b0:1;    /* bit 0 */
};
```

```
#define PMR1    *(volatile unsigned char *)0xFFE0 /* Port mode register 1 */
#define PCR1    *(volatile unsigned char *)0xFFE4 /* Port control register 1 */
#define PDR1    *(volatile unsigned char *)0xFFD4 /* Port data register 1 */

#define PDR2    *(volatile unsigned char *)0xFFD5 /* Port data register 2 */
#define PCR2    *(volatile unsigned char *)0xFFE5 /* Port control register 2 */
#define PDR2_BIT (*(struct BIT *)0xFFD5)
#define SCLK    PDR2_BIT.b0 /* MAX1408 SCLK */
#define DOUT    PDR2_BIT.b2 /* MAX1408 DOUT */
#define DIN     PDR2_BIT.b1 /* MAX1408 DIN */

#define PMR5    *(volatile unsigned char *)0xFFE1 /* Port mode register 5 */
#define PUCR5    *(volatile unsigned char *)0xFFD1 /* Port pull-up control register 5 */
#define PDR5    *(volatile unsigned char *)0xFFD8 /* Port data register 5 */
#define PCR5    *(volatile unsigned char *)0xFFE8 /* Port control register 5 */

#define PDR7    *(volatile unsigned char *)0xFFDA /* Port data register 7 */
#define PCR7    *(volatile unsigned char *)0xFFEA /* Port control register 7 */
#define PDR7_BIT (*(struct BIT *)0xFFDA)
#define CS      PDR7_BIT.b4 /* MAX1408 CS */
```

```
#define WU1      PDR7_BIT.b5          /* MAX1408 WU1 */
#define WU2      PDR7_BIT.b6          /* MAX1408 WU2 */

#define PDR8     *(volatile unsigned char *)0xFFDB /* Port data register 8 */
#define PCR8     *(volatile unsigned char *)0xFFEB /* Port control register 8 */
#define PDR8_BIT (*(struct BIT *)0xFFDB)
#define PWRON    PDR8_BIT.b4          /* MAX1408 POWER */

#define TMRW     *(volatile unsigned char *)0xFF80 /* Timer mode register W */
#define TCRW     *(volatile unsigned char *)0xFF81 /* Timer control register W */
#define TCRW_BIT (*(struct BIT *)0xFF81)          /* Timer Control Register W */
#define TIERW    *(volatile unsigned char *)0xFF82 /* Timer interrupt enable register W */
#define TIERW_BIT (*(struct BIT *)0xFF82)          /* Timer Interrupt Enable Register */
#define OVIEW    TIERW_BIT.b7          /* Timer Overflow Interrupt Enable W */
#define TSRW     *(volatile unsigned char *)0xFF83 /* Timer status register W */
#define TSRW_BIT (*(struct BIT *)0xFF83)          /* Timer Status Register W */
#define OVFW     TSRW_BIT.b7          /* Timer Over flow W */

#define TCRV0    *(volatile unsigned char *)0xFFA0 /* Timer control register V0 */
#define TCRV0_BIT (*(struct BIT *)0xFFA0)
#define OVIE     TCRV0_BIT.b5          /* Timer overflow interrupt enable */
#define TCSRVS   *(volatile unsigned char *)0xFFA1 /* Timer control/status register V */
#define TCSRVS_BIT (*(struct BIT *)0xFFA1)
#define OVFL     TCSRVS_BIT.b5          /* Timer overflow flag */
#define TCRV1    *(volatile unsigned char *)0xFFA5 /* Timer control register V1 */

#pragma interrupt (tmrw)
#pragma interrupt (tmrv)

/* function definition */
extern void INIT(void);          /* Stack pointer set */
void main(void);                 /* main routine */
void tmrw(void);                 /* Timer W interrupt routine */
void tmrv(void);                 /* Timer V interrupt routine */
void ADC_Init(void);             /* ADC initialize */
void DataOut(int ByteNo);        /* ADC write */
void DataIn(int ByteNo);         /* ADC read */

/* Data table */
const unsigned char dsp_data[16] =
{
    0x3f,          /* LED display data = "0" */
    0x06,          /* LED display data = "1" */
    0x5b,          /* LED display data = "2" */
    0x4f,          /* LED display data = "3" */
    0x66,          /* LED display data = "4" */
    0x6d,          /* LED display data = "5" */
    0x7d,          /* LED display data = "6" */
    0x27,          /* LED display data = "7" */
    0x7f,          /* LED display data = "8" */
    0x6f,          /* LED display data = "9" */
    0x77,          /* LED display data = "A" */
    0x7c,          /* LED display data = "b" */
    0x39,          /* LED display data = "C" */

```

```

0x5e,          /* LED display data = "d" */
0x79,          /* LED display data = "E" */
0x71          /* LED display data = "F" */
};

/* RAM define */
unsigned char dig_0;          /* Dig-0 LED display data store */
unsigned char dig_1;          /* Dig-1 LED display data store */
unsigned char dig_2;          /* Dig-2 LED display data store */
unsigned char dig_3;          /* Dig-3 LED display data store */
unsigned char cnt;            /* LED enable counter */
unsigned char counter_sub;
unsigned char exec_flag;      /* exec flag */

volatile unsigned char  sdata[5];          /* send data area */
volatile unsigned char  rdata[5];          /* receive data area */

/* Vector address */
#pragma section V1              /* Vector section set */
void (*const VEC_TBL1[]) (void) = {
    INIT                          /* H'0000 Reset vector */
};
#pragma section V2              /* Vector section set */
void (*const VEC_TBL2[]) (void) = {
    tmrw                          /* H'002a Timer W interrupt vector */
};
#pragma section V3              /* Vector section set */
void (*const VEC_TBL3[]) (void) = {
    tmrv                          /* H'002c Timer V interrupt vector */
};
#pragma section                /* P */

/*****
/* Main program */
*****/
void main(void)
{
    int i;

    set_imask_ccr(1);          /* CCR I-bit = 1 */

    dig_0 = 0x40;              /* Used RAM area initialize */
    dig_1 = 0x40;              /* Used RAM area initialize */
    dig_2 = 0x40;              /* Used RAM area initialize */
    dig_3 = 0x40;              /* Used RAM area initialize */
    cnt   = 0x00;              /* Used RAM area initialize */

    PMR1 = 0x00;               /* Port 1 initialize */
    PCR1 = 0x3f;

    PCR2  = 0xFD;              /* Port 2 initialize */

    PMR5  = 0x00;              /* Port 5 initialize */
    PUCR5 = 0x00;

```

```

PDR5  = 0x00;
PCR5  = 0xff;

PCR7  = 0xff;                                /* Port 7 initialize */

PCR8  = 0x10;                                /* Port 8 initialize */

CS = 1;                                     /* CS High */
PWRON = 0;                                  /* POWER OFF */
WU1 = 1;                                    /* Wake up 1 High */
WU2 = 1;                                    /* Wake up 2 High */

ADC_Init();                                /* ADC initialize */

/* Timer W initialize */
TCRW = 0x20;                                /* Clock Select */
TSRW = 0x00;                                /* Clear OVF */
TMRW = 0x80;                                /* Timer Counter Count Start */
TIERW = 0x00;                                /* OVF Interrupt Disable */
counter_sub = 15;                            /* Initialize 8bit Counter_sub */

TCRV0 = 0x03;                                /* Timer V initialize */
TCRV1 = 0xe2;                                /* Internal clock select */
TCSRv = 0x10;                                /* Clear OVF to 0 */

OVF = 0;                                     /* Clear OVF to 0 */
OVIE = 1;                                    /* Timer V OVF interrupt enable */
OVFW = 0;                                    /* Clear OVF to 0 */
OVIEW = 1;                                   /* Timer W OVF interrupt enable */

set_imask_ccr(0);                            /* CCR I-bit = 0 */

while(1);
}

/*****
/* Timer W Interrupt */
*****/
void tmrw(void)
{
    if ( OVFW == 1 ) {
        OVFW = 0;                            /* Clear OVF */
        counter_sub--;                        /* Decrement 8bit Counter */
        if ( counter_sub == 0x00 ){           /* 8bit Counter != H'00 */
            counter_sub = 15;                /* Initialize 8bit Counter_sub */

            rdata[0] = 0x00;
            do {
                sdata[0] = 0xcc;              /* Read to Status Register */
                DataIn(1);
            } while((rdata[0] & 0x02) == 0);
            sdata[0] = 0xc4;                  /* Read DATA Register value */
            DataIn(2);
        }
    }
}

```



```

        dig_0 = dsp_data[rdata[1] & 0x0f];          /* Dig-0 LED display data set */
        dig_1 = dsp_data[rdata[1] >> 4 & 0x0f];      /* Dig-1 LED display data set */
        dig_2 = dsp_data[rdata[0] & 0x0f];          /* Dig-2 LED display data set */
        dig_3 = dsp_data[rdata[0] >> 4 & 0x0f];      /* Dig-3 LED display data set */
    }
}

/*****
/* Timer V Interrupt
*****/
void tmrv(void)
{
    unsigned char *ptr;                          /* Pointer set */

    ptr = &dig_0;                                /* LED display data store address set*/

    while(OVF == 1){                             /* OVF = 1 ? */
        OVF = 0;                                /* Clear OVF to 0 */
        ptr += cnt;                             /* LED display data read */
        PDR5 = *ptr;                             /* LED display data output */
        PDR1 = cnt << 4;                         /* LED enable data output */
        cnt++;                                  /* "cnt" increment */
        if (cnt >= 4){                           /* 4 times end ? */
            cnt = 0;                             /* "cnt" initialize */
        }
    }
}

/*****
/* ADC initialize function
*****/
void ADC_Init(void)
{
    long i;

    PWRON = 1;                                  /* POWER ON */

    for(i=0;i<10000;i++);                       /* delay */
    WU1 = 0;

    sdata[0] = 0xba;                             /* Write RUN Register */
    DataOut(1);

    sdata[0] = 0xb0;                             /* Write POWER1 Register */
    sdata[1] = 0xf0;
    DataOut(2);

    sdata[0] = 0x82;                             /* Write MUX Register */
    sdata[1] = 0x61;
    DataOut(2);

    sdata[0] = 0x80;                             /* Write ADC Register */
    sdata[1] = 0x11;

```

```

    DataOut(2);
}

/*****
/* ADC register write and read function */
*****/
void DataIn(int ByteNo)
{
    int cSclk, Dcnt, maxdcnt, i, Pcnt;
    unsigned char BitPos;

    rdata[0] = rdata[1] = 0x00;          /* data area initialize */

    maxdcnt = (ByteNo << 4);
    Dcnt = Pcnt = 0;                    /* counter initialize */
    BitPos = 0x80;                      /* bit position initialize */
    CS = 0;                             /* CS ON */
    SCLK = cSclk = 0;                   /* SCLK Low */
    if(sdata[Dcnt] & BitPos)            /* output data */
        DOUT = 1;
    else
        DOUT = 0;
    BitPos >>= 1;                        /* next bit position */

    /* write register */
    while(1){
        for(i=0;i<10;i++);              /* delay */
        cSclk ^= 1;
        SCLK = cSclk;                   /* output reverse SCLK */
        if(Pcnt & 0x01) {
            if((Pcnt % 16) != 15){
                if(sdata[Dcnt] & BitPos) /* last bit ? */
                    DOUT = 1;           /* output data */
                else
                    DOUT = 0;
                BitPos >>= 1;            /* next bit position */
            } else {
                for(i=0;i<10;i++);      /* delay */
                SCLK = 0;               /* SCLK Low */
                DOUT = 1;               /* DOUT Low */
                break;
            }
        }
        Pcnt++;                          /* pulse count increment */
    }

    /* read register */
    Dcnt = Pcnt = 0;                    /* SCLK count */
    BitPos = 0x80;                      /* bitb position initialize */
    while(1){
        if((Pcnt & 0x01) == 0x00) {
            if(DIN)
                rdata[Dcnt] |= BitPos; /* input data */
            BitPos >>= 1;               /* next bit position */
        }
    }
}

```

```

        if(Pcnt >= (maxdcnt - 1)) {           /* end ?           */
            for(i=0;i<10;i++);                /* delay               */
            SCLK = 0;                          /* SCLK Low            */
            for(i=0;i<10;i++);                /* delay               */
            CS = 1;                            /* CS High             */
            return;
        } else if(((Pcnt % 16) == 0) && (Pcnt != 0)){ /* next data          */
            BitPos = 0x80;                     /* bitb position initialize */
            Dcnt++;                             /* data count increment   */
            if(DIN)                             /* input data            */
                rdata[Dcnt] |= BitPos;
            BitPos >>= 1;                       /* next bit position      */
        }
    }
    cSclk ^= 1;
    SCLK = cSclk;                             /* reverse SCLK          */
    Pcnt++;                                    /* pulse count increment  */
    for(i=0;i<10;i++);                        /* delay                 */
}

/*****
/* ADC register write function
*****/
void DataOut(int ByteNo)
{
    int cSclk, Dcnt, maxdcnt, i, Pcnt;
    unsigned char BitPos;

    maxdcnt = (ByteNo << 4);
    Dcnt = Pcnt = 0;                          /* counter initialize */
    BitPos = 0x80;                             /* bit position initialize */
    CS = 0;                                    /* CS ON              */
    SCLK = cSclk = 0;                          /* SCLK Low           */
    if(sdata[Dcnt] & BitPos)                   /* output data */
        DOUT = 1;
    else
        DOUT = 0;
    BitPos >>= 1;                              /* next bit position */

    while(1){
        for(i=0;i<10;i++);                    /* delay              */
        cSclk ^= 1;
        SCLK = cSclk;                          /* reverse SCLK        */
        if(Pcnt & 0x01) {
            if((Pcnt % 16) != 15){             /* not last bit ?     */
                if(sdata[Dcnt] & BitPos)       /* output data */
                    DOUT = 1;
                else
                    DOUT = 0;
                BitPos >>= 1;                   /* next bit position */
            } else {
                if(Pcnt >= (maxdcnt - 1)) {     /* end ?              */
                    for(i=0;i<10;i++);         /* delay              */

```

```
SCLK = 0;                /* SCLK Low */
DOUT = 1;                /* output High */
for(i=0;i<10;i++);      /* delay */
CS = 1;                 /* CS OFF */
return;
} else {
    BitPos = 0x80;       /* bit position initialize */
    Dcnt++;             /* data count increment */
    if(sdata[Dcnt] & BitPos) /* output data */
        DOUT = 1;
    else
        DOUT = 0;
    BitPos >>= 1;        /* next bit position */
}
}
}
Pcnt++;                /* pulse count increment */
}
}
```

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Dec.20.03	—	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.