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

Connecting a semiconductor-type acceleration sensor

Introduction

An acceleration sensor is connected to the analog input pin AN0, and fractional acceleration values (units: g) are displayed on 7-segment LEDs.

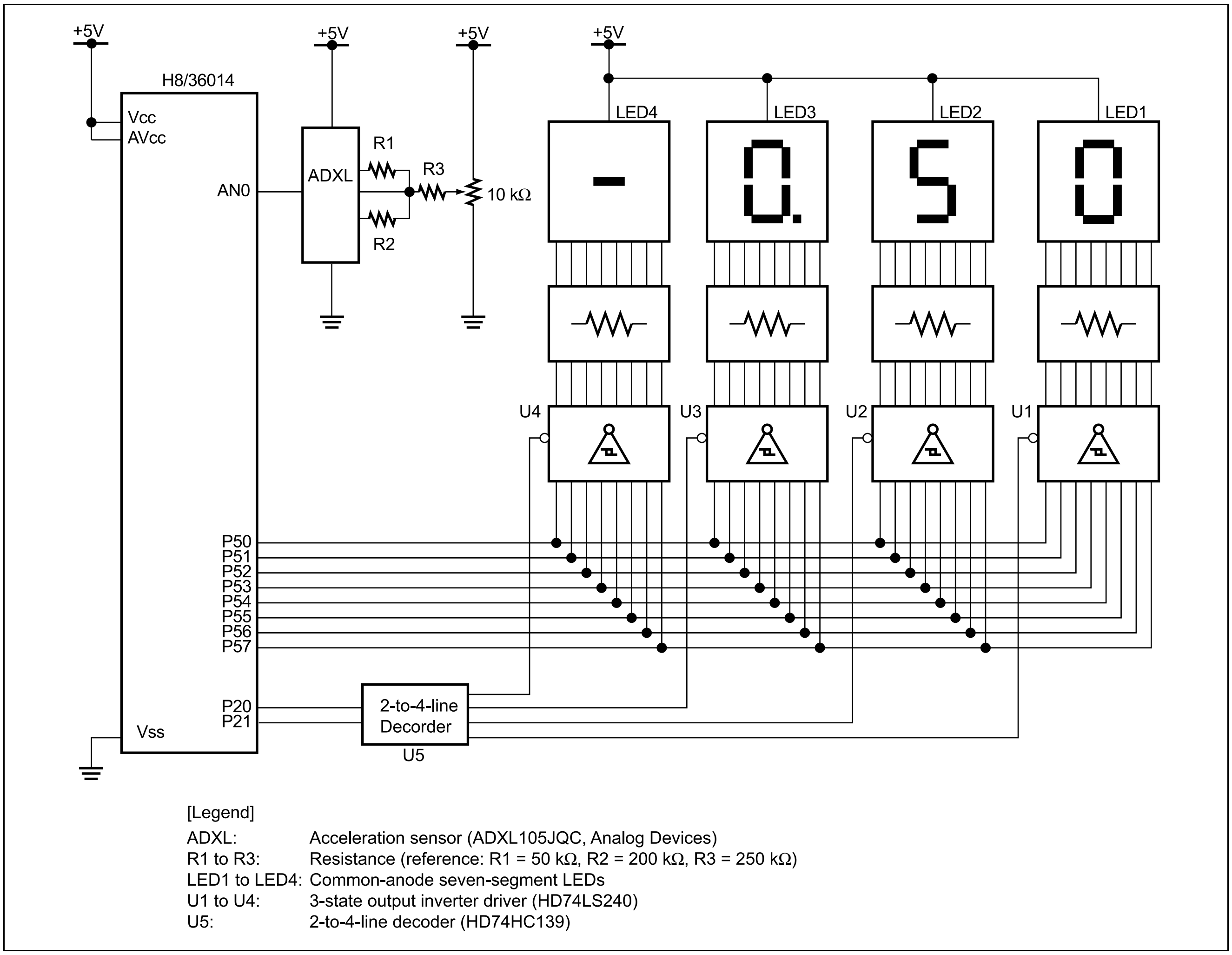
Target Device

H8/36014

Contents

1. Specifications

1. Figure 1.1 shows the hardware configuration of an example of semiconductor type acceleration sensor connection. As shown in the figure, the sensor is connected to the analog input pin 0 (pin AN0).
2. A/D conversion of the AN0 pin input is performed, and the A/D conversion result is displayed on seven-segment LEDs connected to an I/O port.
3. The seven-segment LED displays show the 10-bit data of the A/D conversion result in decimal acceleration.
4. The A/D conversion is performed every 0.49152 seconds.



[Legend]

ADXL:	Acceleration sensor (ADXL105JQC, Analog Devices)
R1 to R3:	Resistance (reference: R1 = 50 kΩ, R2 = 200 kΩ, R3 = 250 kΩ)
LED1 to LED4:	Common-anode seven-segment LEDs
U1 to U4:	3-state output inverter driver (HD74LS240)
U5:	2-to-4-line decoder (HD74HC139)

Figure 1.1 Hardware Configuration

5. In this sample task, the H8/36014 operating voltage (Vcc) and analog power supply voltage (AVcc) are 5.0 V, and the OSC clock frequency is 16 MHz.

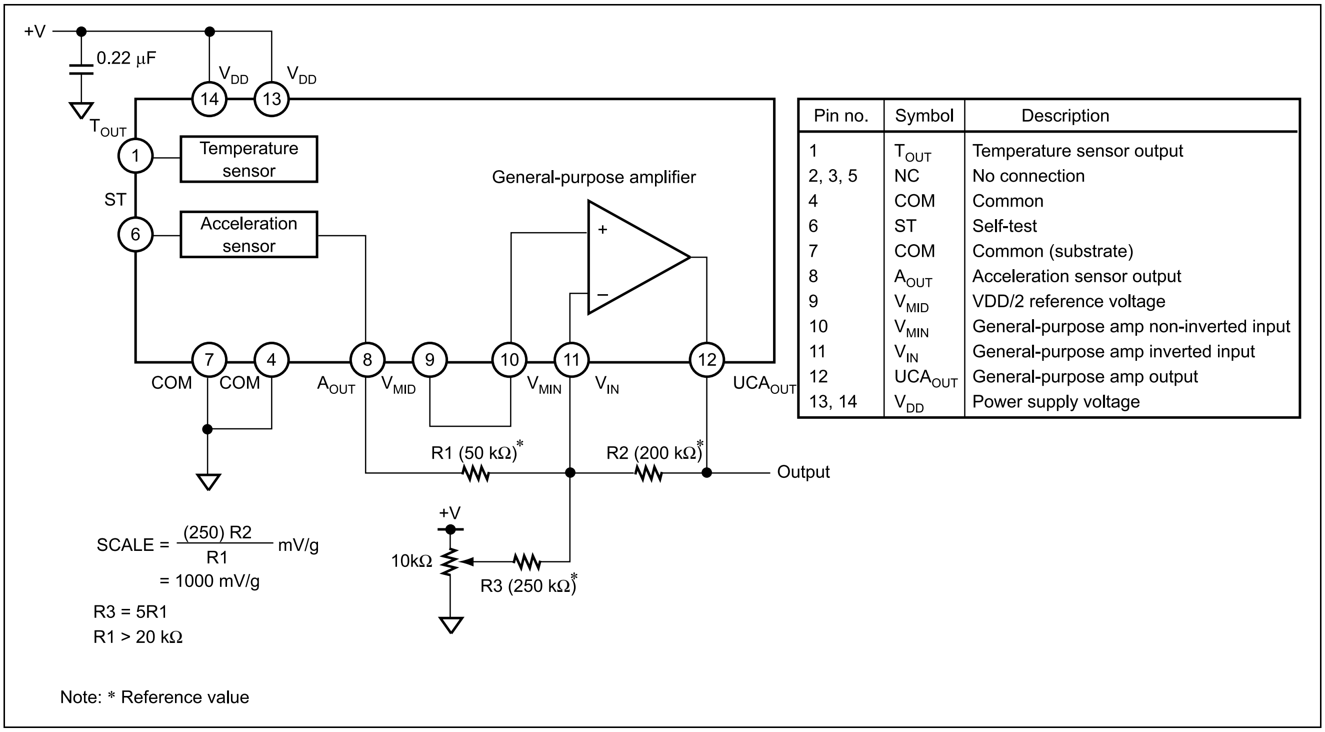
6. The semiconductor type acceleration sensor used in this task example is a high-precision single-axis acceleration sensor having an analog input, and manufactured by Analog Devices (model no. ADXL105JQC). Specifications are shown below.

   A. Table 1.1 shows specifications of the semiconductor type acceleration sensor.

Table 1.1 ADXL105JQC specifications (reference)

		ADXL105J/A			
Parameter	**Conditions**	**Min.**	**Typ.**	**Max.**	**Units**
Sensitivity (proportional)	Value at Aout	225	250	275	mV/g
Initial	Vs = 2.7 V	90	105	120	mV/g
Zero-g bias level	Value at Aout	–625		+625	mV
Zero-g offset error	From nominal value + 2.5 V				
Power supply		2.70		5.25	V
Operating voltage range					

   B. Figure 1.2 shows an application reference circuit.



Pin no.	Symbol	Description
1	T_{OUT}	Temperature sensor output
2, 3, 5	NC	No connection
4	COM	Common
6	ST	Self-test
7	COM	Common (substrate)
8	A_{OUT}	Acceleration sensor output
9	V_{MID}	VDD/2 reference voltage
10	V_{MIN}	General-purpose amp non-inverted input
11	V_{IN}	General-purpose amp inverted input
12	UCA_{OUT}	General-purpose amp output
13, 14	V_{DD}	Power supply voltage

$\text{SCALE} = \frac{(250)\,R2}{R1}$ mV/g

= 1000 mV/g

R3 = 5R1

R1 > 20 kΩ

Note: * Reference value

Figure 1.2 Application circuit with increased scale factor (reference)

   C. The ADXL105 is capable of measuring both dynamic acceleration (characteristic of vibrations) and static acceleration (for example, inertial forces, gravity and tilt (inclination)). In this task, the internal general-purpose amplifier is used to change the scale factor.

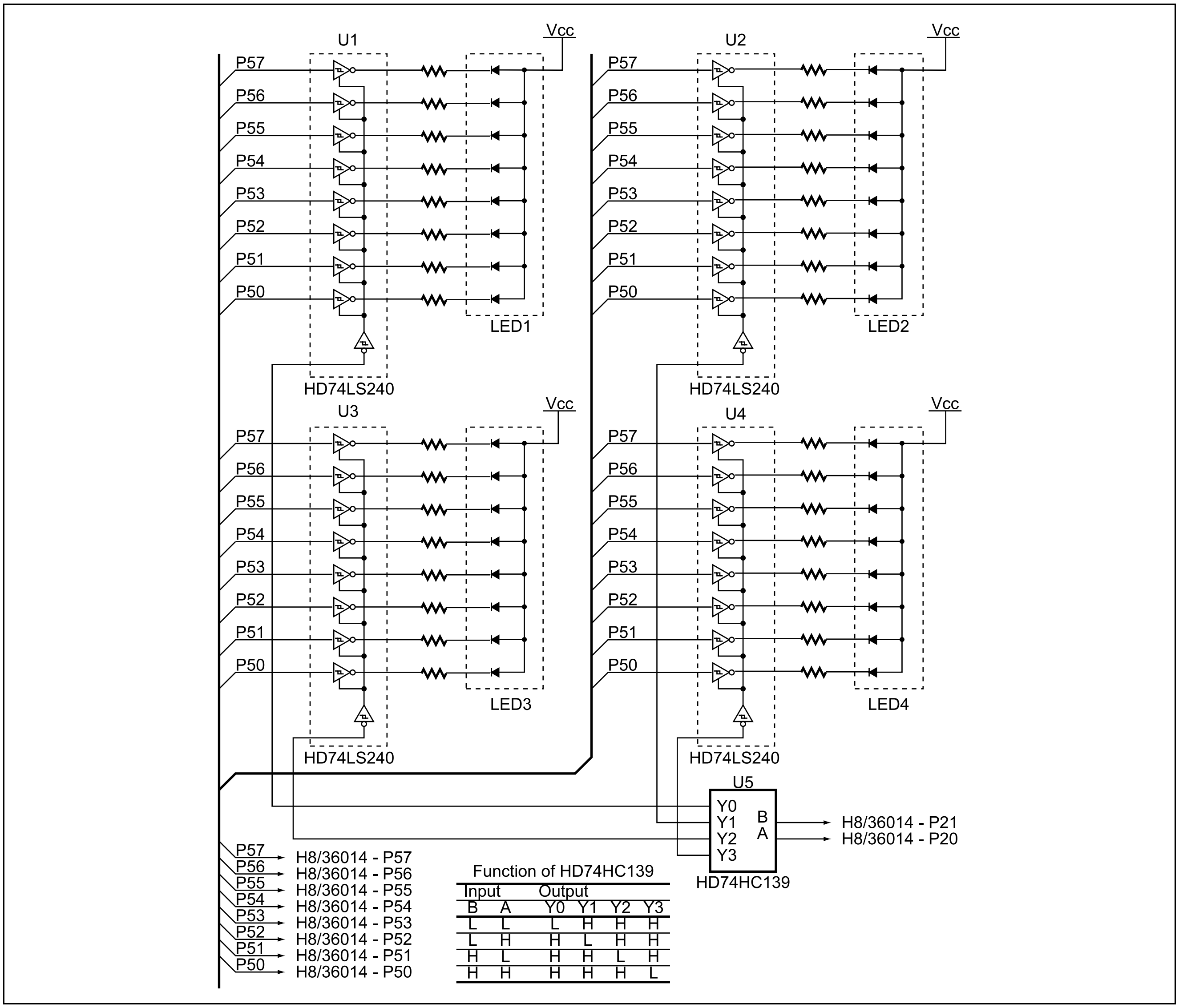
$$UCA_{OUT} = V_{DD} / 2 + \text{SCALE} \times g$$

[Legend]

UCA_{OUT}:	General-purpose amp output
V_{DD}:	Power supply voltage
SCALE:	Scale (mV/g)
g:	Acceleration (g)

7. Operation in this task example is as follows.
   A. The ADXL105 is set to a nominal scale factor of 250 mV/g (V_{DD} = 5 V), but the gain is changed to 1000 mV/g (V_{DD} = 5 V) using an external resistance. (By changing the gain and adjusting the 0g level, the measurement range changes.)
   B. If dynamic acceleration and static acceleration are brought to bear on the acceleration sensor, the acceleration (in units of g, gravitational acceleration) is displayed on the LEDs accordingly.
   C. If the sensor is positioned vertically with the index (semicircular cutout) downward, +1 g is measured as the static acceleration, and the LED display reads "1.00". If positioned vertically with the index upward, –1 g is measured, and the LED display reads "–1.00".

8. In this sample task, in order to display results on seven-segment LEDs, the port output is input to three-state output inverter drivers (HD74LS240), and the driver outputs are connected to the cathodes of the seven-segment LEDs. The port used for display on four seven-segment LEDs is connected to all of the seven-segment LEDs; switching of the display on the seven-segment LEDs is controlled by the enable pin of the three-state inverter drivers. The 2-to-4-line decoder (HD74HC139) is used to generate signals for switching the seven-segment LED display, and is controlled by two port outputs. Figure 1.3 shows the method of control of the seven-segment LEDs.



Function of HD74HC139

Input		Output			
B	A	Y0	Y1	Y2	Y3
L	L	L	H	H	H
L	H	H	L	H	H
H	L	H	H	L	H
H	H	H	H	H	L

Figure 1.3 Method of Control of Seven-Segment LEDs

9. In this task example, the acceleration value (units: g) is displayed to two decimal places on 7-segment LEDs. Figure 1.4 shows the method by which the acceleration value is displayed on the LEDs.

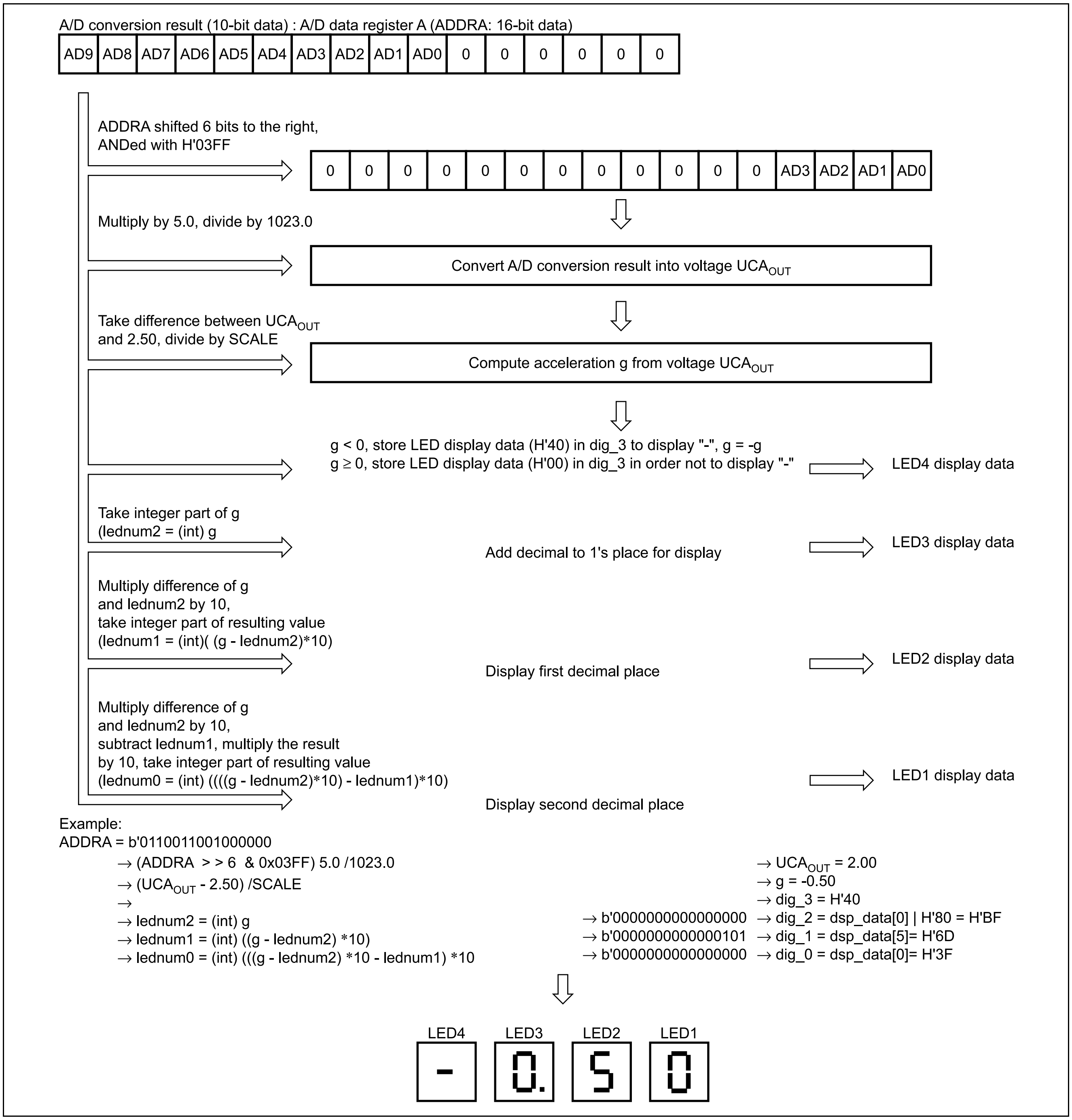

A/D conversion result (10-bit data) : A/D data register A (ADDRA: 16-bit data)
AD9 AD8 AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0 0 0 0 0 0 0
ADDRA shifted 6 bits to the right,
ANDed with H'03FF
0 0 0 0 0 0 0 0 0 0 0 0 AD3 AD2 AD1 AD0
Multiply by 5.0, divide by 1023.0
Convert A/D conversion result into voltage UCA_OUT
Take difference between UCA_OUT and 2.50, divide by SCALE
Compute acceleration g from voltage UCA_OUT
g < 0, store LED display data (H'40) in dig_3 to display "-", g = -g
g ≥ 0, store LED display data (H'00) in dig_3 in order not to display "-"
LED4 display data
Take integer part of g
(lednum2 = (int) g
Add decimal to 1's place for display
LED3 display data
Multiply difference of g
and lednum2 by 10,
take integer part of resulting value
(lednum1 = (int)((g - lednum2)*10)
Display first decimal place
LED2 display data
Multiply difference of g
and lednum2 by 10,
subtract lednum1, multiply the result
by 10, take integer part of resulting value
(lednum0 = (int) ((((g - lednum2)*10) - lednum1)*10)
Display second decimal place
LED1 display data
Example:
ADDRA = b'0110011001000000
→ (ADDRA > > 6 & 0x03FF) 5.0 /1023.0
→ (UCA_OUT - 2.50) /SCALE
→
→ lednum2 = (int) g
→ lednum1 = (int) ((g - lednum2) *10)
→ lednum0 = (int) (((g - lednum2) *10 - lednum1) *10
→ UCA_OUT = 2.00
→ g = -0.50
→ dig_3 = H'40
→ b'0000000000000000 → dig_2 = dsp_data[0] | H'80 = H'BF
→ b'0000000000000101 → dig_1 = dsp_data[5]= H'6D
→ b'0000000000000000 → dig_0 = dsp_data[0]= H'3F
LED4
LED3
LED2
LED1
-
0.
5
0

Figure 1.4 Acceleration value is displayed on the LEDs.

2. Description of Functions

Figure 2.1 is a block diagram of the H8/36014 functions used in this sample task; table 2.1 shows function allocations.

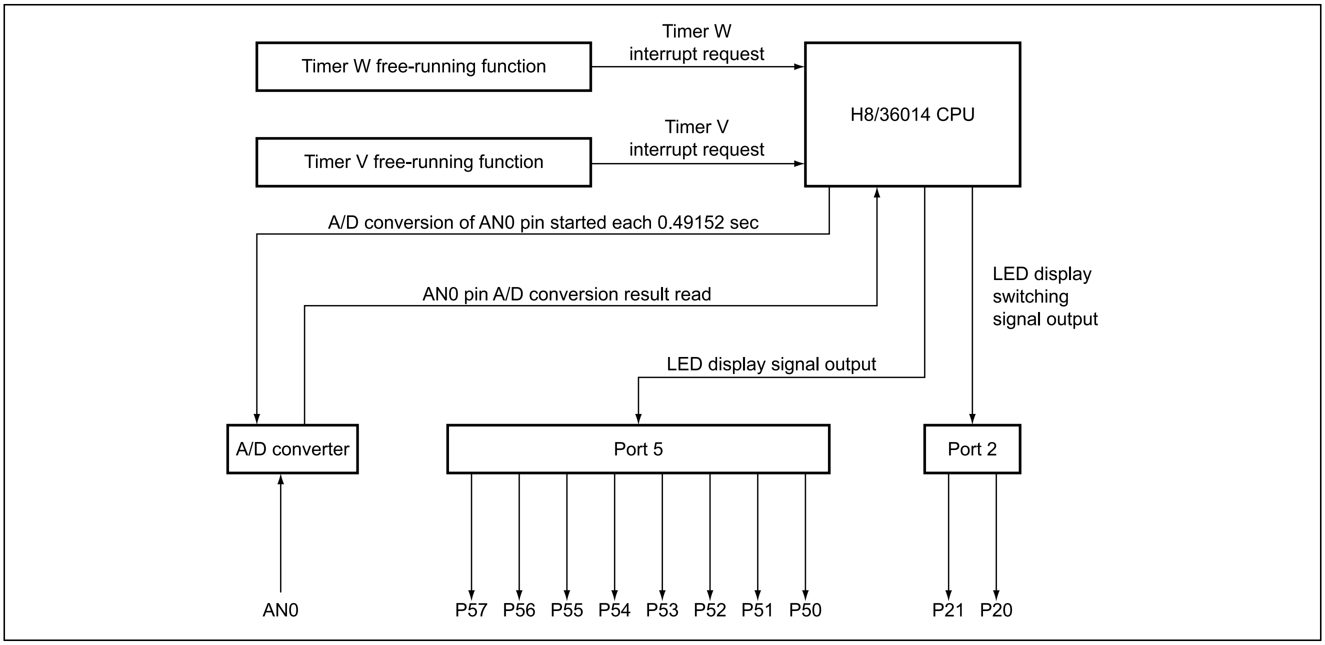

Timer W free-running function
Timer W interrupt request
H8/36014 CPU
Timer V free-running function
Timer V interrupt request
A/D conversion of AN0 pin started each 0.49152 sec
AN0 pin A/D conversion result read
LED display switching signal output
LED display signal output
A/D converter
Port 5
Port 2
AN0
P57
P56
P55
P54
P53
P52
P51
P50
P21
P20

Figure 2.1 Block Diagram of Functions Used

Table 2.1 Function Allocation

Function Used	Function Allocation
Timer W	The timer W free running function is used to perform A/D conversion of the analog input pin 0 (AN0) signal. The A/D conversion period is measured by the timer W overflow period of 32.768 ms.
Timer V	The timer V free running function is used to control switching of the 7-segment LED display. Dynamic lighting is performed by lighting the four 7-segment LEDs in order at every 2.048 ms Timer V overflow period.
A/D converter	The output voltage from the acceleration sensor connected to the A/D converter analog input pin 0 (AN0) is A/D converted.
Port 2	Display on the four 7-segment LEDs is switched by the port 2 output pins P20 and P21. The P20 and P21 output pins are connected to I/O pins of the 2-to-4-line decoder.
Port 5	Results are displayed on the 7-segment LEDs by output from the port 5 output pins P50 to P57. The 10-bit data which is the A/D conversion result at pin AN0 is converted into a decimal acceleration display data (unit: g) for output to the LEDs.

2. A diagram of the connection of the seven-segment LEDs used is shown in figure 2.2. As shown in the figure, by outputting "high" from port 5, the corresponding LED segments are lit. The relation between the port 5 outputs and the LED display data appears in table 2.2.

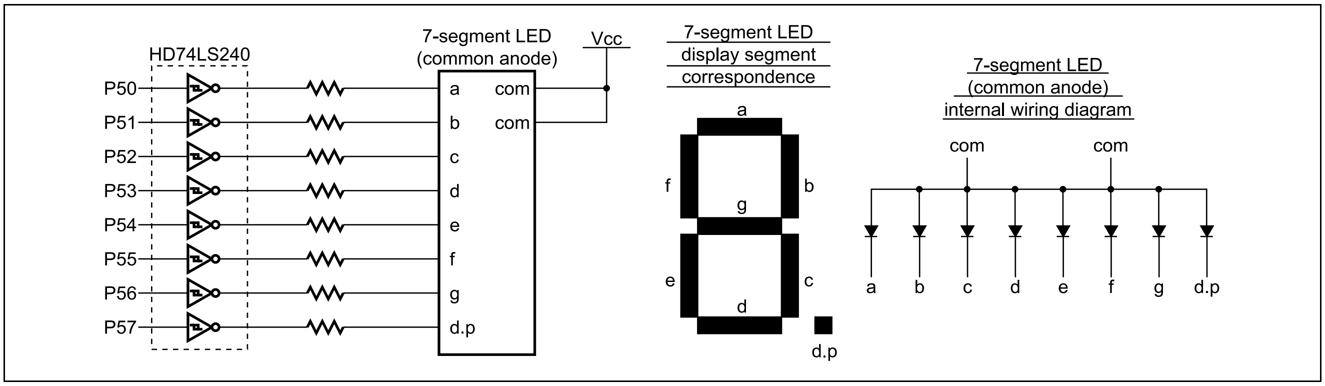
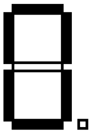
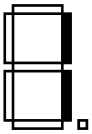
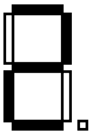
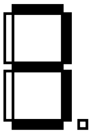
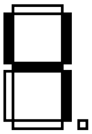
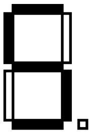
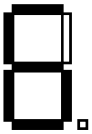
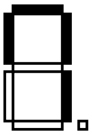
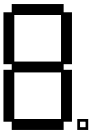
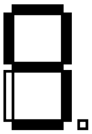
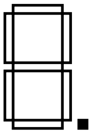
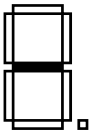


Figure 2.2 7-Segment LED Connection Diagram and Internal Wiring

Table 2.2 Relation between Port 5 Outputs and 7-Segment LED Display Data

LED Display	Port 5 Output Data							
	P57	P56	P55	P54	P53	P52	P51	P50
	0	0	1	1	1	1	1	1
	0	0	0	0	0	1	1	0
	0	1	0	1	1	0	1	1
	0	1	0	0	1	1	1	1
	0	1	1	0	0	1	1	0
	0	1	1	0	1	1	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
	1	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0

Note: The first integer digit is ANDed with a decimal point.

3. Principles of Operation

1. Figure 3.1 shows the principle of operation when using the timer W to perform A/D conversion of the AN0 pin output. As shown in figure 3.1, in this sample task the A/D conversion period (0.49152 s) is measured using the timer W overflow flag in the tmrw routine, without using the A/D conversion interrupt, to judge the end of A/D conversion.

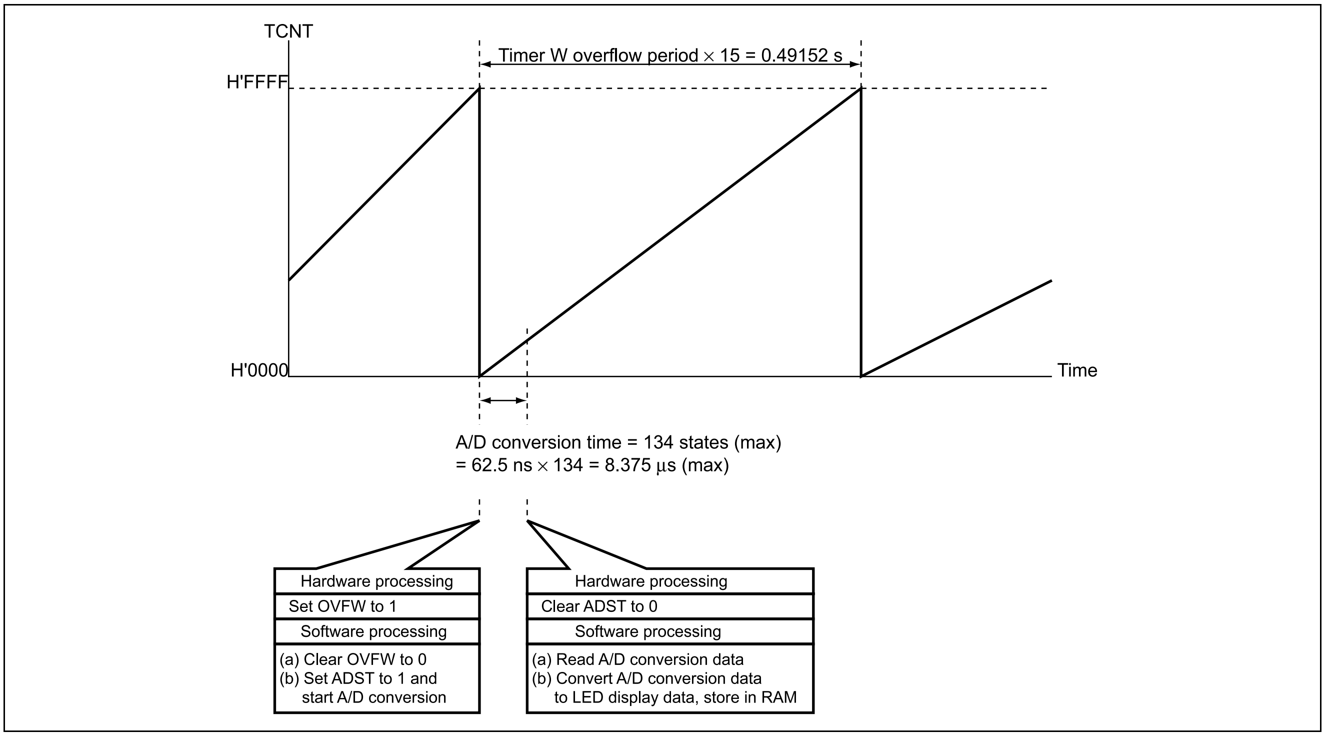

TCNT
Timer W overflow period × 15 = 0.49152 s
H'FFFF
H'0000
Time
A/D conversion time = 134 states (max)
= 62.5 ns × 134 = 8.375 µs (max)
Hardware processing
Set OVFW to 1
Software processing
(a) Clear OVFW to 0
(b) Set ADST to 1 and start A/D conversion
Hardware processing
Clear ADST to 0
Software processing
(a) Read A/D conversion data
(b) Convert A/D conversion data to LED display data, store in RAM

Figure 3.1 Principle of Operation of A/D Conversion of the AN0 Pin Output using Timer W

2. The principle of operation of the seven-segment LED display control is as follows. Figure 3.2 describes the operation for the case in which "–0.50" is displayed on LED4 through LED1. As shown in figure 3.2, by displaying the data in order on LED1 through LED4 at each timer V overflow period, data is displayed dynamically on the seven-segment LEDs.

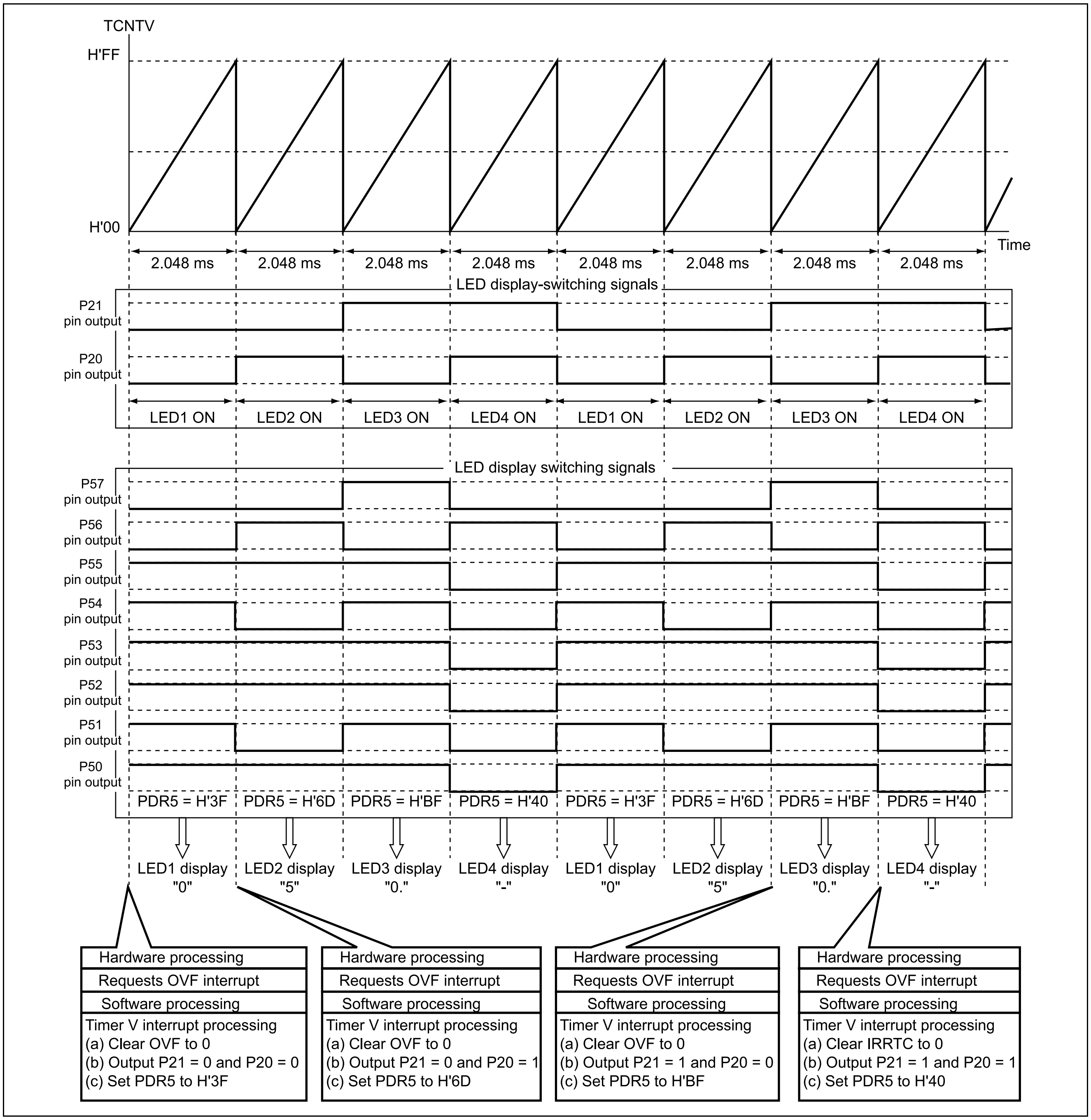


Figure 3.2 Principle of Operation of 7-Segment LED Display Control

4. Description of Software

4.1 Description of Modules

Table 4.1 describes the modules used in this sample task.

Table 4.1 Description of Modules

Module Name	Label Name	Function
Main routine	main	Sets initial values, enables interrupts
Timer W interrupt processing routine	tmrw	Clears interrupt flag, converts A/D conversion data into LED display data, and stores in RAM
Timer V interrupt processing routine	tmrv	Clears interrupt flag and controls LED display data output and LED display switching

4.2 Description of arguments

This sample task does not use arguments.

4.3 Description of Internal Registers

The internal registers used in this sample task are described in table 4.2.

Table 4.2 Description of Internal Registers

Register Name		Function	Address	Setting
TCRV0		Timer control register V0: Selects TCNTV input clock, sets TCNTV clear conditions, and controls interrupt requests	H'FFA0	H'03 (initial setting)
	CMIEB	Compare-match interrupt enable B: when CMIEB = 0, disables interrupt requests by CMFB in TCSRV	Bit 7	0
	CMIEA	Compare-match interrupt enable A: when CMIEA = 0, disables interrupt requests by CMFA in TCSRV	Bit 6	0
	OVIE	Timer overflow interrupt enable: when OVIE = 0, disables interrupt requests by OVF in TCSRV, when OVIE = 1, enables interrupt requests by OVF in TCSRV	Bit 5	0/1
	CCLR1	Counter clear 1, 0: Sets conditions to clear TCNTV When CCLR1 = 0 and CCLR0 = 0, TCNTV clearing is disabled	Bit 4	0
	CCLR0		Bit 3	0
	CKS2	Clock select 2 to 0: Combined with ICKS0 in TCRV1, selects clock for input to TCNTV and count conditions When CKS2 = 0, CKS1 = 1, CKS0 = 1, and ICKS0 = 1, TCNTV counts on the falling edge of the internal clock ϕ/128	Bit 2	0
	CKS1		Bit 1	1
	CKS0		Bit 0	1

Register Name		Function	Address	Setting
TCSRV		Timer control/status register V: Displays status flag, controls output of compare-match	H'FFA1	H'10
	CMFB	Compare-match flag B: Set to 1 when the values of TCNTV and TCORB match	Bit 7	0
	CMFA	Compare-match flag A: Set to 1 when the values of TCNTV and TCORA match	Bit 6	0
	OVF	Timer overflow flag: When the TCNTV value overflows, OVF is set to 1 In the OVF = 1 state, when 0 is written to OVF after reading OVF, OVF is cleared to 0	Bit 5	0
	OS3	Output select 3, 2: Sets the output level of the TMOV pin for the output of compare-match B When set to OS3 = 0 and OS2 = 0, no change occurs	Bit 3	0
	OS2		Bit 2	0
	OS1	Output select 1, 0: Sets the output level of the TMOV pin for the output of compare-match A When set to OS1 = 0 and OS0 = 0, no change occurs	Bit 1	0
	OS0		Bit 0	0
TCRV1		Timer control register V1: Selects TRGV pin edge select, TRGV input enable, TCNTV input clock	H'FFA5	H'E3
	TVEG1	TRGV input edge select 1, 0: Selects input edge of TRGV pin When set to TREG1 = 0, TREG0 = 0, disables trigger input from TRGV pin	Bit 4	0
	TVEG0		Bit 3	0
	TRGE	TRGV input enable: Enables/disables TCNTV count-up by edge input selected using TVEG1 and TVEG0 When set to TREG = 0, disables start of TCNTV count-up due to input to TRGV pin and halt of TCNTV count-up when TCNTV is cleared due to a compare-match	Bit 2	0
	ICKS0	Internal clock select 0: Combined with CKS2 to CKS0 of TCRV0, selects clock for input to TCNTV and count conditions; when set to CKS2 = 0, CKS1 = 1, CKS0 = 1 and ICKS0 = 1, TCNTV counts on the falling edge of the internal clock ϕ/128	Bit 0	1
TMRW		Timer mode register W: Selects general register functions and timer output mode	H'FF80	H'80
	CTS	Counter start: When CTS = 1, indicates TCNT has started counting When CTS = 0, indicates TCNT has stopped counting	Bit 7	1
TCRW		Timer control register W: Selects counter clock Sets counter clear conditions and timer output level	H'FF81	H'30
	CKS2	Clock select: When CKS2 = 0, CKS1 = 1 and CKS0 = 1, sets TCNT input clock to system clock divided by 8	Bit 6	0
	CKS1		Bit 5	1
	CKS0		Bit 4	1
TIERW		Timer interrupt enable register W: controls timer W interrupt requests	H'FF82	H'00 (initial setting)
	OVIE	Timer overflow interrupt enable: when OVIE = 0, disables interrupt requests by OVF When OVIE = 1, enables interrupt requests by OVF	Bit 7	0/1

Register Name		Function	Address	Setting
TSRW		Indicates interrupt request status	H'FF83	H'00
	OVF	Timer overflow: When OVF = 0, indicates TCNT has not overflowed When OVF = 1, indicates that TCNT has overflowed	Bit 7	0
TCNT		Timer counter: 16-bit count-up counter which takes as input the system clock frequency-divided by 8	H'FF86	H'00
ADCSR		A/D control/status register: Consists of A/D converter control bits and conversion status bits	H'FFB8	H'00 (initial setting)
	ADF	A/D end flag: Set to 1 when A/D conversion ends in single mode After reading in the ADF = 1 state, ADF is cleared to 0 by writing 0	Bit 7	0
	ADIE	A/D interrupt enable: When 0, disables A/D conversion end interrupt requests by the ADF	Bit 6	0
	ADST	A/D start: Setting this to 1 starts A/D conversion Automatically cleared to 0 when A/D conversion ends in single mode	Bit 5	0/1
	SCAN	Scan mode: When 0, A/D conversion mode is single mode	Bit 4	0
	CKS	Clock select: When 0, A/D conversion time = 134 states (max)	Bit 3	0
	CH2	Channel select: Selects the analog input channel; when CH2 = 0, CH1 = 0 and CH0 = 0 (SCAN = 0), input channel is AN1	Bit 2	0
	CH1		Bit 1	0
	CH0		Bit 0	0
ADCR		A/D control register: Enables A/D conversion start by an external trigger	H'FFB9	H'7E
	TRGE	Trigger enable: When 0, disables the start of A/D conversion by edge input to the external trigger pin (ADTRG)	Bit 7	0
ADDRA		A/D data register A: Stores the AN0 A/D conversion result	H'FFB0	—
PCR2		Port control register 2: Selects, for each bit, pin I/O to use as the port 2 general I/O port When PCR2 = H'03, pin P22 functions as a general input pin, and pins P21 and P20 function as general output pins	H'FFE5	H'03
PDR2		Port data register 2: General I/O port data register for port 2	H'FFD5	H'F8
PMR5		Port mode register 5: Sets the port 5 pin functions	H'FFE1	H'00
	POF7	P57 pin function switch: When 0, P57 functions as a general I/O port	Bit 7	0
	POF6	P56 pin function switch: When 0, P56 functions as a general I/O port	Bit 6	0
	WKP5	P55/$\overline{\text{WKP5}}$/$\overline{\text{ADTRG}}$ pin function switch: When 0, P55 functions as a general I/O port	Bit 5	0
	WKP4	P54/$\overline{\text{WKP4}}$ pin function switch: When 0, P54 functions as a general I/O port	Bit 4	0
	WKP3	P53/$\overline{\text{WKP3}}$ pin function switch: When 0, P53 functions as a general I/O port	Bit 3	0
	WKP2	P52/$\overline{\text{WKP2}}$ pin function switch: When 0, P52 functions as a general I/O port	Bit 2	0
	WKP1	P51/$\overline{\text{WKP1}}$ pin function switch: When 0, P51 functions as a general I/O port	Bit 1	0
	WKP0	P50/$\overline{\text{WKP0}}$ pin function switch: When 0, P50 functions as a general I/O port	Bit 0	0

Register Name	Function	Address	Setting
PUCR5	Port pull-up control register 5: Controls, for each bit, the pull-up MOS of each port 5 pin set to an input port When PUCR5 = H'00, pull-up MOS for pins P57 to P50 is turned off	H'FFD1	H'00
PDR5	Port data register 5: Port 5 general I/O port data register	H'FFD8	H'00
PCR5	Port control register 5: Selects, for each bit, the pin I/O used as a port 5 general I/O port When PCR5 = H'FF, the pins P57 to P50 function as general output pins	H'FFE8	H'FF

4.4 Description of RAM

Table 4.3 describes the RAM used in this sample task.

Table 4.3 Description of RAM

Label Name	Function	Address	Used in
dig_0	Store LED1 display data (1 byte)	H'FB94	main, tmrw
dig_1	Store LED2 display data (1 byte)	H'FB95	main, tmrw
dig_2	Store LED3 display data (1 byte)	H'FB96	main, tmrw
dig_3	Store LED4 display data (1 byte)	H'FB97	main, tmrw
UCA_{OUT}	Voltage value	H'FB80	tmrw
g	Acceleration value	H'FB84	tmrw
SCALE	Scale factor	H'FB88	main, tmrw
lednum0	LED1 display data	H'FB8C	tmrw
lednum1	LED2 display data	H'FB8E	tmrw
lednum2	LED3 display data	H'FB90	tmrw
*ptr	Pointer to store dig_0 address	H'FB92	tmrv
cnt	8-bit counter to switch LED1 to LED4 display (1 byte)	H'FB98	main, tmrv
counter_sub	8-bit counter to adjust A/D acquisition interval (1 byte)	H'FB99	main, tmrw

4.5 Description of data table

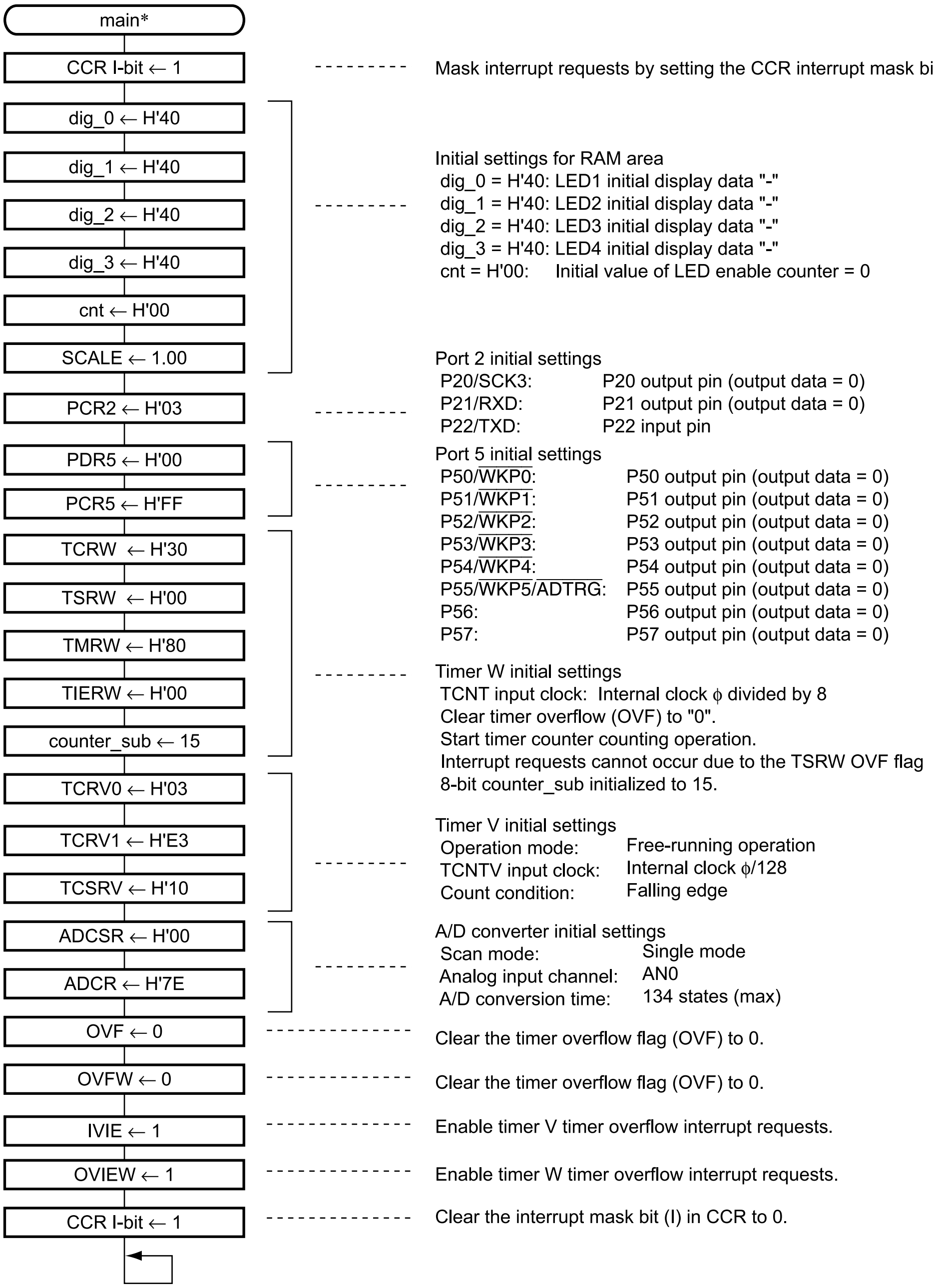
In this sample task, seven-segment LED display data is stored as a one-dimensional data table in ROM. Table 4.4 describes the seven-segment LED display data table (dsp_data[ ]).

Table 4.4 Description of 7-Segment LED Display Data Table (dsp_data[])

Array Name	Data	Data Description	Data Size	Address
dsp_data[0]	H'3F	Data output from port 5 to display "0" on an LED	1 byte	H'1116
dsp_data[1]	H'06	Data output from port 5 to display "1" on an LED	1 byte	H'1117
dsp_data[2]	H'5B	Data output from port 5 to display "2" on an LED	1 byte	H'1118
dsp_data[3]	H'4F	Data output from port 5 to display "3" on an LED	1 byte	H'1119
dsp_data[4]	H'66	Data output from port 5 to display "4" on an LED	1 byte	H'111A
dsp_data[5]	H'6D	Data output from port 5 to display "5" on an LED	1 byte	H'111B
dsp_data[6]	H'7D	Data output from port 5 to display "6" on an LED	1 byte	H'111C
dsp_data[7]	H'27	Data output from port 5 to display "7" on an LED	1 byte	H'111D
dsp_data[8]	H'7F	Data output from port 5 to display "8" on an LED	1 byte	H'111E
dsp_data[9]	H'6F	Data output from port 5 to display "9" on an LED	1 byte	H'111F

5. Flowchart

1. Main routine (main)



Mask interrupt requests by setting the CCR interrupt mask bit (I) to 1.

Initial settings for RAM area
dig_0 = H'40: LED1 initial display data "-"
dig_1 = H'40: LED2 initial display data "-"
dig_2 = H'40: LED3 initial display data "-"
dig_3 = H'40: LED4 initial display data "-"
cnt = H'00: Initial value of LED enable counter = 0

Port 2 initial settings
P20/SCK3: P20 output pin (output data = 0)
P21/RXD: P21 output pin (output data = 0)
P22/TXD: P22 input pin

Port 5 initial settings
P50/$\overline{\text{WKP0}}$: P50 output pin (output data = 0)
P51/$\overline{\text{WKP1}}$: P51 output pin (output data = 0)
P52/$\overline{\text{WKP2}}$: P52 output pin (output data = 0)
P53/$\overline{\text{WKP3}}$: P53 output pin (output data = 0)
P54/$\overline{\text{WKP4}}$: P54 output pin (output data = 0)
P55/$\overline{\text{WKP5}}$/$\overline{\text{ADTRG}}$: P55 output pin (output data = 0)
P56: P56 output pin (output data = 0)
P57: P57 output pin (output data = 0)

Timer W initial settings
TCNT input clock: Internal clock ϕ divided by 8
Clear timer overflow (OVF) to "0".
Start timer counter counting operation.
Interrupt requests cannot occur due to the TSRW OVF flag
8-bit counter_sub initialized to 15.

Timer V initial settings
Operation mode: Free-running operation
TCNTV input clock: Internal clock ϕ/128
Count condition: Falling edge

A/D converter initial settings
Scan mode: Single mode
Analog input channel: AN0
A/D conversion time: 134 states (max)

Clear the timer overflow flag (OVF) to 0.

Clear the timer overflow flag (OVF) to 0.

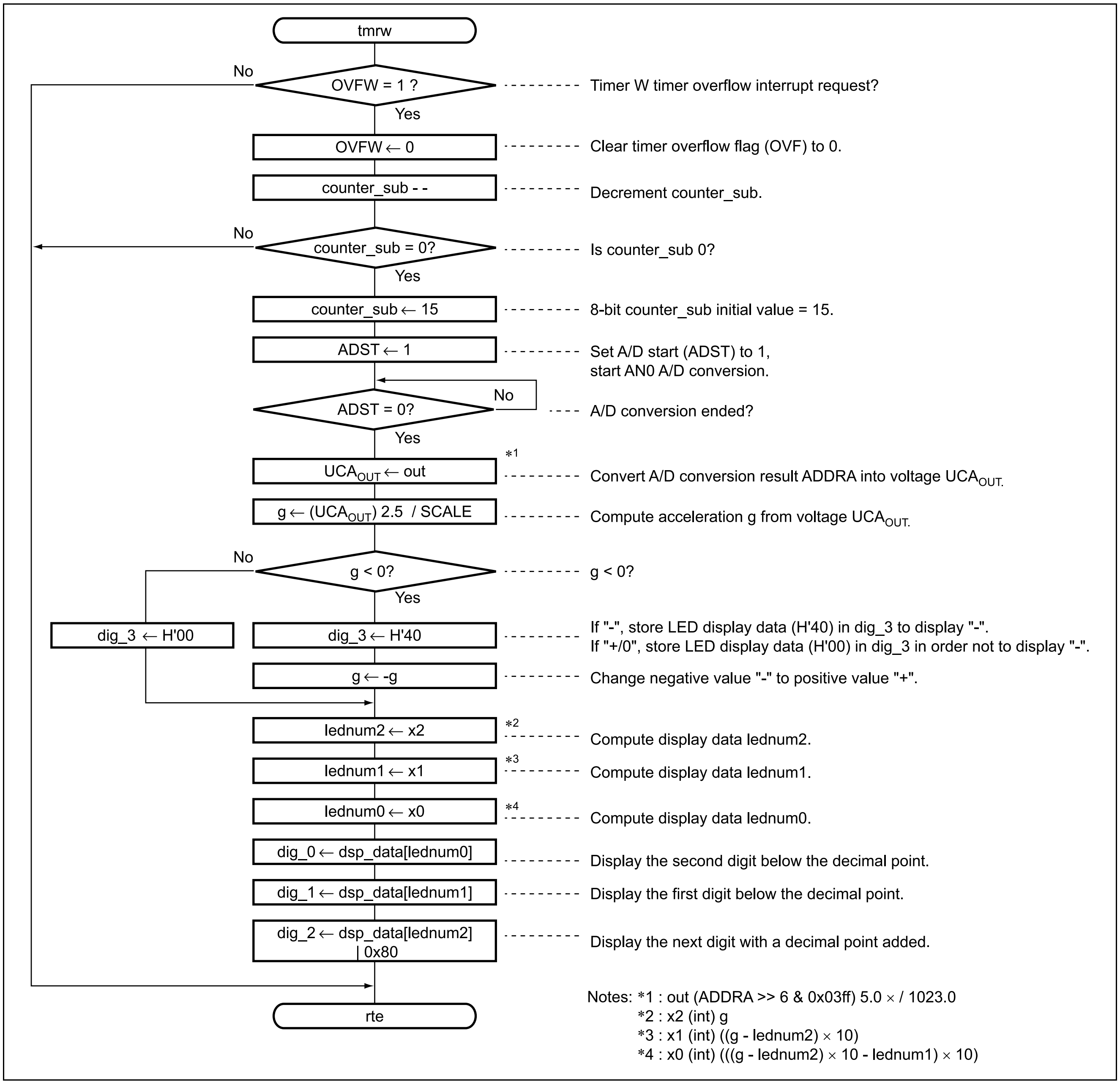
Enable timer V timer overflow interrupt requests.

Enable timer W timer overflow interrupt requests.

Clear the interrupt mask bit (I) in CCR to 0.

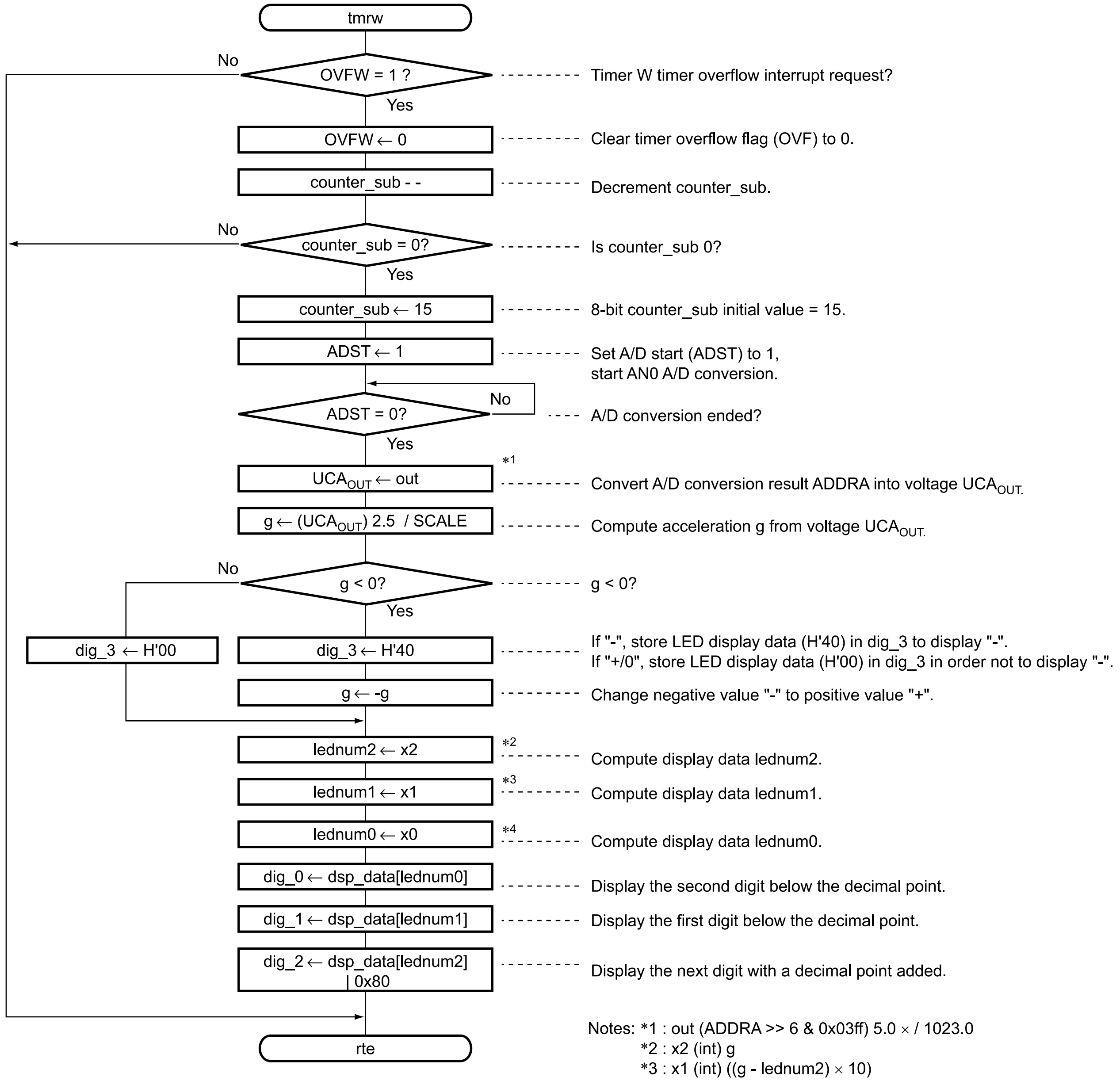
Note: * In this sample task, the stack pointer is specified by INIT.SRC (assembly language).

2. Timer W interrupt processing routine (tmrw)


tmrw
No
OVFW = 1 ?
Timer W timer overflow interrupt request?
Yes
OVFW ← 0
Clear timer overflow flag (OVF) to 0.
counter_sub - -
Decrement counter_sub.
No
counter_sub = 0?
Is counter_sub 0?
Yes
counter_sub ← 15
8-bit counter_sub initial value = 15.
ADST ← 1
Set A/D start (ADST) to 1,
start AN0 A/D conversion.
No
ADST = 0?
A/D conversion ended?
Yes
*1
UCA_OUT ← out
Convert A/D conversion result ADDRA into voltage UCA_OUT.
g ← (UCA_OUT) 2.5 / SCALE
Compute acceleration g from voltage UCA_OUT.
No
g < 0?
g < 0?
Yes
dig_3 ← H'00
dig_3 ← H'40
If "-", store LED display data (H'40) in dig_3 to display "-".
If "+/0", store LED display data (H'00) in dig_3 in order not to display "-".
g ← -g
Change negative value "-" to positive value "+".
lednum2 ← x2
*2
Compute display data lednum2.
lednum1 ← x1
*3
Compute display data lednum1.
lednum0 ← x0
*4
Compute display data lednum0.
dig_0 ← dsp_data[lednum0]
Display the second digit below the decimal point.
dig_1 ← dsp_data[lednum1]
Display the first digit below the decimal point.
dig_2 ← dsp_data[lednum2]
| 0x80
Display the next digit with a decimal point added.
rte

Notes: *1 : out (ADDRA >> 6 & 0x03ff) 5.0 × / 1023.0
*2 : x2 (int) g
*3 : x1 (int) ((g - lednum2) × 10)
*4 : x0 (int) (((g - lednum2) × 10 - lednum1) × 10)

3. Timer V interrupt processing routine (tmrv)



Timer W timer overflow interrupt request?

Clear timer overflow flag (OVF) to 0.

Decrement counter_sub.

Is counter_sub 0?

8-bit counter_sub initial value = 15.

Set A/D start (ADST) to 1,
start AN0 A/D conversion.

A/D conversion ended?

Convert A/D conversion result ADDRA into voltage $UCA_{OUT.}$

Compute acceleration g from voltage $UCA_{OUT.}$

g < 0?

If "-", store LED display data (H'40) in dig_3 to display "-".
If "+/0", store LED display data (H'00) in dig_3 in order not to display "-".

Change negative value "-" to positive value "+".

Compute display data lednum2.

Compute display data lednum1.

Compute display data lednum0.

Display the second digit below the decimal point.

Display the first digit below the decimal point.

Display the next digit with a decimal point added.

Notes: *1 : out (ADDRA >> 6 & 0x03ff) 5.0 × / 1023.0
*2 : x2 (int) g
*3 : x1 (int) ((g - lednum2) × 10)
*4 : x0 (int) (((g - lednum2) × 10 - lednum1) × 10)

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                                                                             */
/*   Semiconductor type acceleration sensor connection example                                       */

#include <machine.h>

/*   Symbol defnition                                                                                */
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 PDR2   *(volatile unsigned char *)0xFFD5                          /* Port data register 2            */
#define PCR2   *(volatile unsigned char *)0xFFE5                          /* Port control register 2         */

#define PDR5        *(volatile unsigned char *)0xFFD8                     /* Port data register 5            */
#define PCR5        *(volatile unsigned char *)0xFFE8                     /* Port control register 5         */
#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 TCSRV       *(volatile unsigned char *)0xFFA1                     /* Timer control/status register V */
#define TCSRV_BIT   (*(struct BIT *)0xFFA1)
#define OVF         TCSRV_BIT.b5                                          /* Timer overflow flag             */
#define TCRV1       *(volatile unsigned char *)0xFFA5                     /* Timer control register V1       */
#define ADCSR       *(volatile unsigned char *)0xFFB8                     /* A/D control/status register     */
#define ADCSR_BIT   (*(struct BIT *)0xFFB8)
#define ADST        ADCSR_BIT.b5                                          /* A/D start                       */
#define ADCR        *(volatile unsigned char *)0xFFB9                     /* A/D control register            */
```

```
#define ADDRA       *(volatile unsigned int *)0xFFB0                                 /* A/D data register A                      */
#define ADDRB       *(volatile unsigned int *)0xFFB2                                 /* A/D data register A                      */
#define ADDRC       *(volatile unsigned int *)0xFFB4                                 /* A/D data register A                      */
#define ADDRD       *(volatile unsigned int *)0xFFB6                                 /* A/D data register A                      */

#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                */

/*   Data table                                                                                                                  */
const unsigned char dsp_data[10] =
{
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"                   */
};

/*   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;
float UCAout;                                                                        /* Output voltage                           */
float g;                                                                             /* Acceleration Scale result (unit:g)       */
float SCALE;                                                                         /* Scale factor for 5.0V                    */
unsigned int lednum0;                                                                /* Dig-0 LED display data                   */
unsigned int lednum1;                                                                /* Dig-1 LED display data                   */
unsigned int lednum2;                                                                /* Dig-2 LED display data                   */
unsigned char *ptr;                                                                  /* Pointer set                              */

/*   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)
{
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                 */
     SCALE = 1.00;                                                                  /* Scale factor for 5V                      */

     PCR2  = 0x03;                                                                  /* Port 2 initialize                        */

     PDR5  = 0x00;                                                                  /* Port 5 initialize                        */
     PCR5  = 0xff;

                                                                                    /* Timer W initialize                       */
     TCRW = 0x30;                                                                   /* 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 = 0xe3;                                                                  /* Internal clock select                    */
     TCSRV = 0x10;                                                                  /* Clear OVF to 0                           */

     ADCSR = 0x00;                                                                  /* A/D converter initialize                 */
     ADCR  = 0x7e;

     OVF     = 0;                                                                   /* Clear OVF to 0                           */
     OVFW    = 0;                                                                   /* Clear OVF to 0                           */
     OVIE    = 1;                                                                   /* Timer V OVF interrupt enable             */
     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              */
                    ADST  = 1;                                                                                                  /*
A/D converter start */
                    while(ADST == 1);                                               /* A/D converter end ?                      */
                    UCAout = (ADDRA >> 6 & 0x03ff) * 5.0 / 1023.0;                  /* Output voltage                           */
                    g = (UCAout - 2.50) / SCALE;                                    /* Acceleration Scale result (unit:g)       */
                    if (g < 0) {                                                    /* Acceleration Scale result < 0            */
                         dig_3 = 0x40;                                              /* Dig-3 LED display data set               */
                         g = -g;                                                    /* g = -g                                   */
                    }else {
                         dig_3 = 0x00;                                              /* Dig-3 LED display data set               */
               }
               lednum2 = (int)g;                                                    /* Compute Dig-2 LED display data           */
               lednum1 = (int)((g - lednum2) * 10);                                 /* Compute Dig-1 LED display data           */
               lednum0 = (int)(((g -lednum2) * 10 -lednum1) * 10);                  /* Compute Dig-0 LED display data           */
               dig_0 = dsp_data[lednum0];                                           /* Dig-0 LED display data set               */
               dig_1 = dsp_data[lednum1];                                           /* Dig-1 LED display data set               */
               dig_2 = dsp_data[lednum2] | 0x80;                                    /* Dig-2 LED display data set               */
          }
     }
}

/********************************************************************************************************************************/
/*   Timer V Interrupt                                                                                                          */
/********************************************************************************************************************************/
void tmrv(void)
{
     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                  */
          PDR2 = cnt;                                                               /* LED enable data output                   */
          cnt++;                                                                    /* "cnt" increment                          */
          if (cnt >= 4){                                                            /* 4 times end ?                            */
               cnt = 0;                                                             /* "cnt" initialize                         */
          }
     }
}
```

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Sep.29.03	—	First edition issued

Keep safety first in your circuit designs!

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

Notes regarding these materials

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