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.

RENESAS

H8/300L SLP Series

Connecting a Semiconductor-Type Acceleration Sensor

Introduction

An acceleration sensor is connected to analog input pin AN0, and the measured acceleration values (unit: g) are displayed in decimal fraction on seven-segment LEDs.

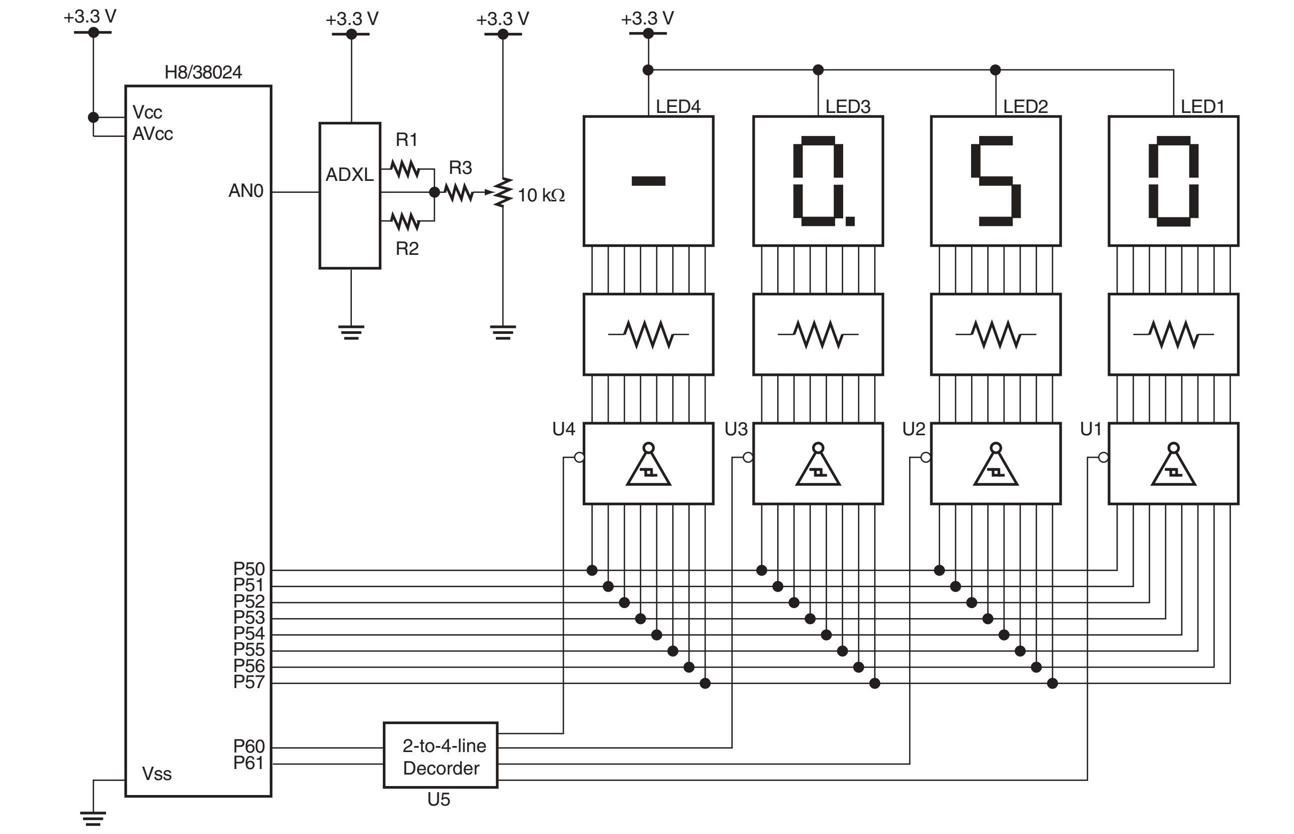
Target Device

H8/38024

Contents

1. Specifications

1. Figure 1.1 shows the hardware configuration of an example of connecting a semiconductor type acceleration sensor. The sensor is connected to analog input pin 0 (pin AN0) as shown in the figure.
2. The signal on the AN0 pin is A/D converted, after which the results of A/D conversions are displayed on the 7-segment LEDs connected to the I/O port.
3. The 7-segment LED display shows the 10-bit result of A/D conversion as an acceleration value in decimal fraction.
4. The A/D conversion is performed at 0.5-s intervals.



[Legend]
ADXL: Acceleration sensor (ADXL105JQC, Analog Devices)
R1 to R3: Resistor (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 H8/38024's operating voltage (Vcc) and analog power supply voltage (AVcc) are 3.3 V, the OSC clock frequency is 10 MHz, and the watch clock is 32.768 kHz.

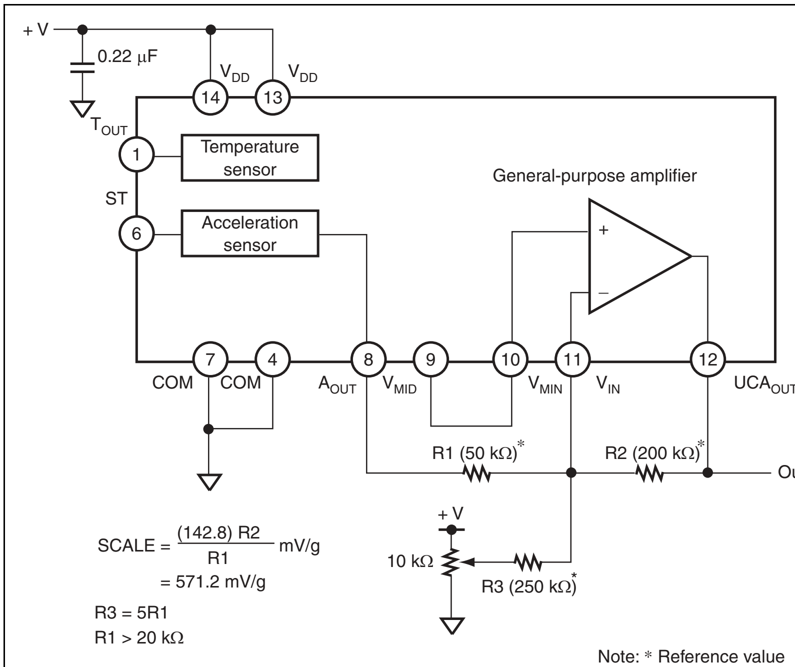
6. The semiconductor type acceleration sensor used in this sample task is a high-accuracy single-axis accelerometer with analog input, which is a product of Analog Devices Inc. (model ADXL105JQC). Its specifications are shown below.
   A. Table 1.1 shows specifications of the semiconductor type acceleration sensor.

Table 1.1 ADXL105JQC Specifications (Reference)

Parameter	Conditions	ADXL105J/A Min.	Typ.	Max.	Units
Sensitivity (proportional)	Value at Aout	117.8	142.8	160.4	mV/g
Initial value	Vs = 2.7 V	90	105	120	mV/g
Zero-g bias level	Value at Aout	–392		+391	mV
Zero-g offset error	From +1.65 V nominal				
Power supply		2.70		5.25	V
Operating voltage range					

Note: Unless otherwise specified, Ta = 25°C and Vs = 3.3V.

   B. Figure 1.2 shows a reference circuit for application.



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

Figure 1.2 Application Circuit with Increased Scale Factor (Reference)

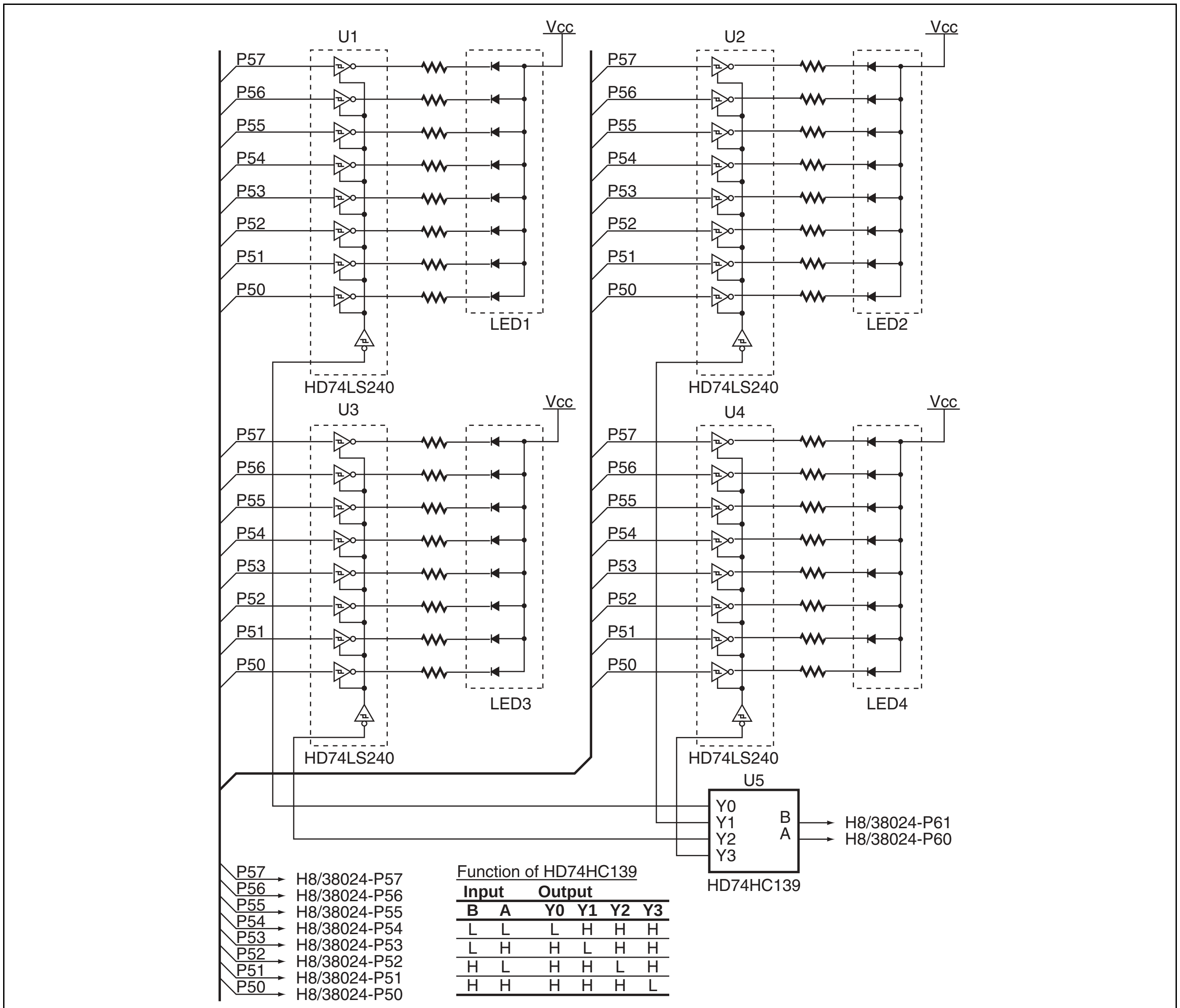
C. The ADXL105 is capable of measuring both dynamic acceleration (characteristic of vibrations) and static acceleration (for example, inertial forces, gravity and tilt (inclination)). In this task, the internal general-purpose amplifier is used to change the scale factor.

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

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

7. Operation of this sample task is as follows.
   A. The ADXL105 is set to a nominal scale factor of 142.8 mV/g (V_{DD} = 3.3 V), but the gain is changed to 571.2 mV/g (V_{DD} = 3.3 V) using an external resistor. (The measurement range changes by changing the gain and adjusting the 0g level.)
   B. If dynamic acceleration and static acceleration are induced 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, display on the 7-segment LED is handled by attaching port outputs to the inputs to the tri-state-output inverter drivers (HD74LS240), and the driver outputs are in turn connected to the cathodes of the 7-segment LEDs. The port outputs are connected to each of the four 7-segment LEDs to control the display on the LEDs. The enable pins of the tri-state inverter driver control switching of display on the 7-segment LEDs. The signals used to switch the display are generated by the 2-to-4-line decoder (HD74HC139), which is controlled by two port-pin outputs. Figure 1.3 shows how the 7-segment LEDs are controlled.



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 Control of 7-Segment LEDs

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

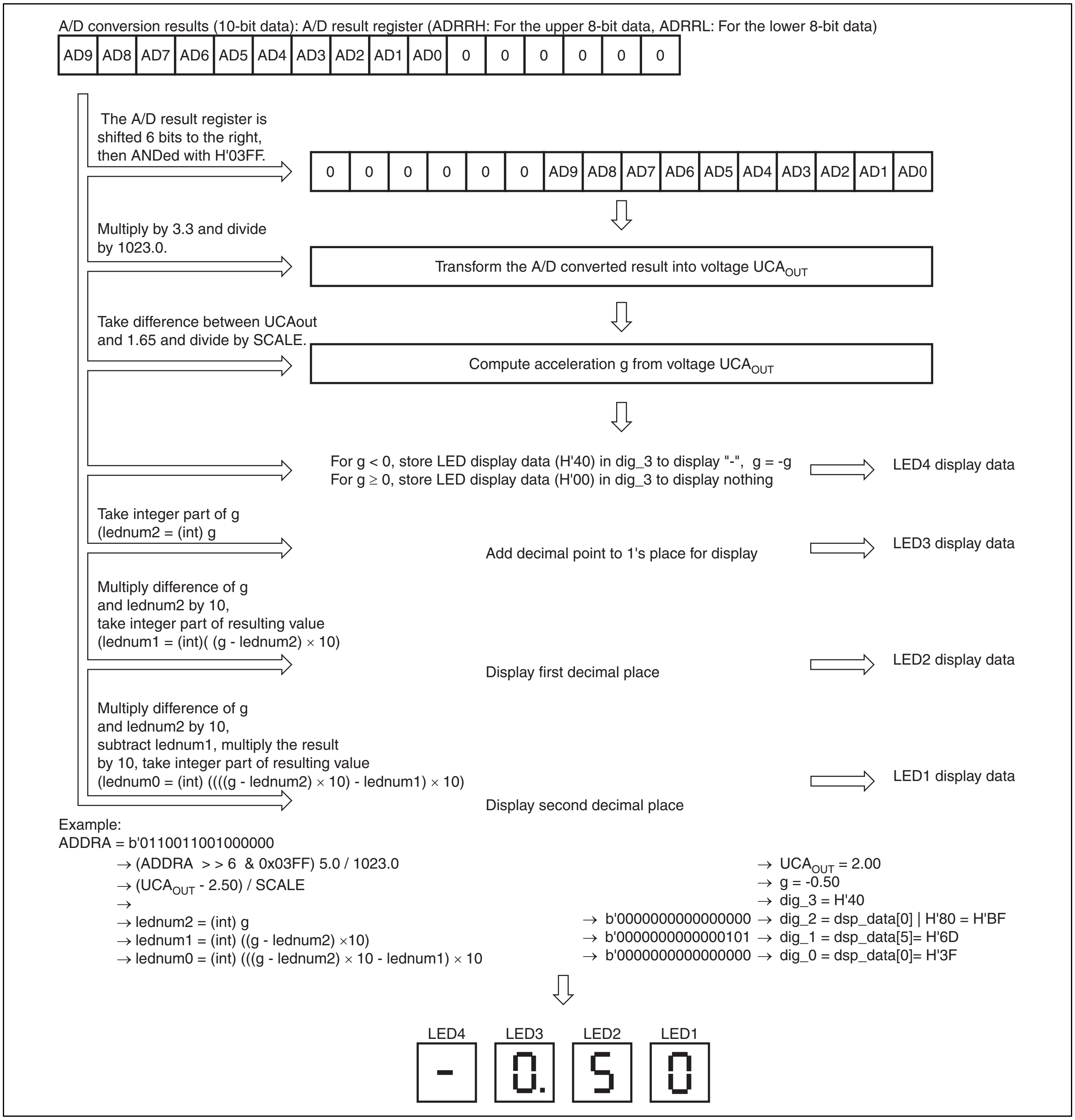


Figure 1.4 How A/D Conversion Results are Displayed on the LEDs

2. Description of Functions

1. Figure 2.1 is a block diagram of the H8/38024 functions used in this sample task. Table 2.1 shows function allocations.

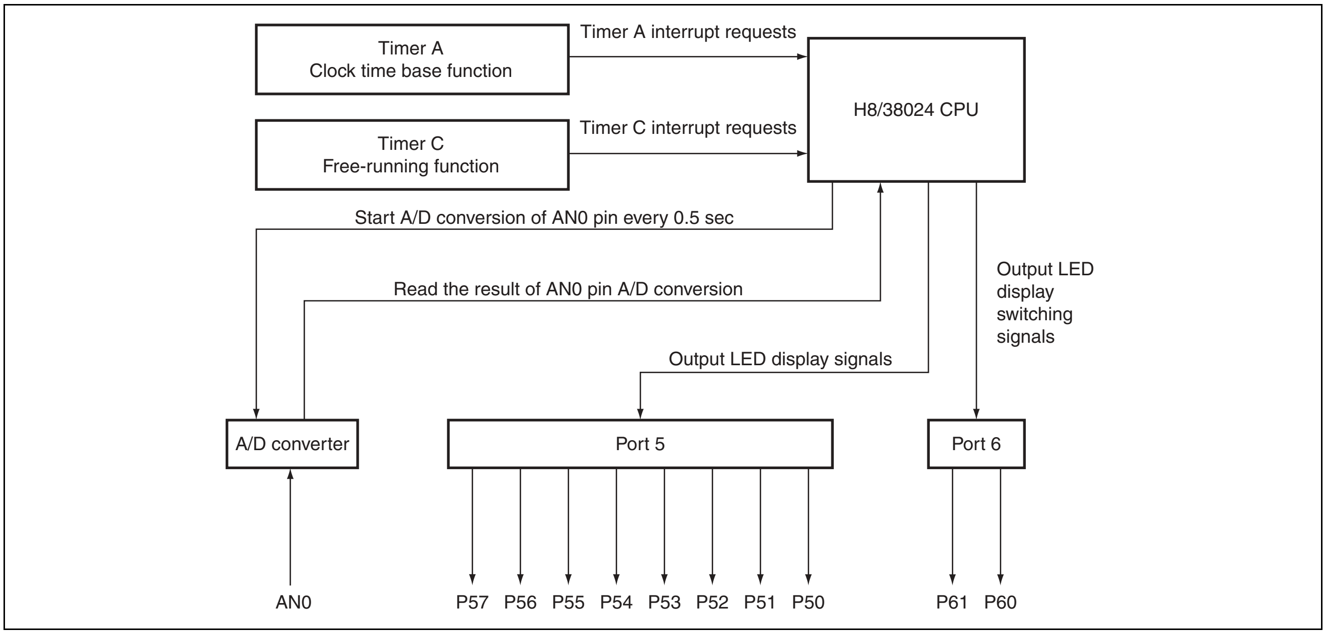


Figure 2.1 Block Diagram of Functions Used

Table 2.1 Function Allocation

Function	Function Allocation
Timer A	The timer A's clock time base function is used to measure 0.5 s, which is the period for A/D conversion of the signal on the analog input pin 0 (AN0). The timer A interrupt is used for each A/D conversion period.
Timer C	Timer C's free-running function is used to control switching of the 7-segment LED display. Each of the four 7-segment LEDs is lit in sequence at an interval of 3.2768 ms, which is the time taken for timer C to overflow. This obtains dynamic illumination from the LEDs.
A/D converter	This unit A/D-converts the output voltage from the acceleration sensor connected to analog input pin 0 (AN0) of the A/D converter.
Port 5	The 7-segment LEDs are displayed by the P50 to P57 output pins of port 5. The 10 bits of data produced by A/D conversion of the value on the AN0 pin are converted to a decimal acceleration data (unit: g) for display, this is then output to the LED.
Port 6	The four 7-segment LED display is switched by the P60 and P61 output pins of port 6. These pins are connected to the input/output pins of the 2-to-4-line decoder.

2. Figure 2.2 shows how the 7-segment LED used in this task is connected. A high output from port 5 lights up the corresponding segment as shown by the figure. Table 2.2 shows the relationship between the output from port 5 and the display on the LED.

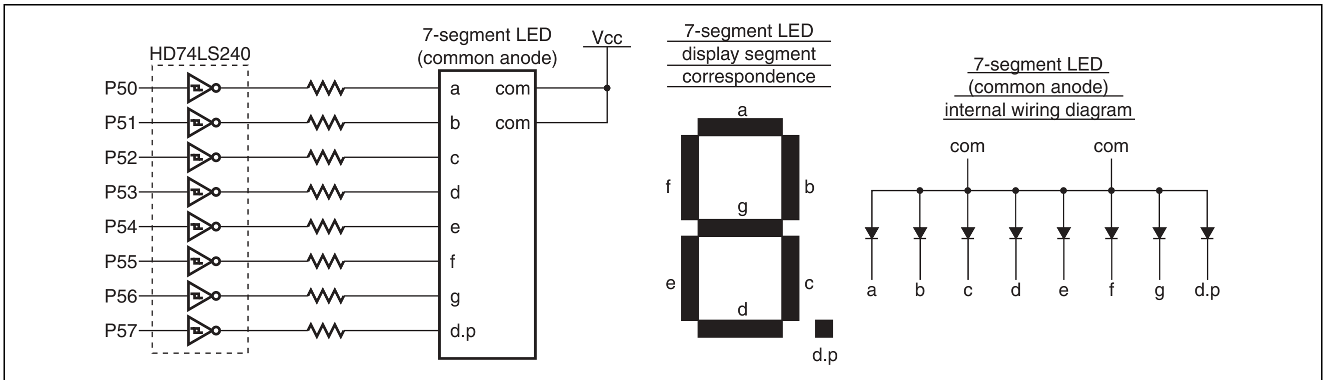
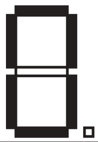
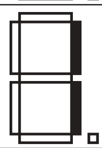
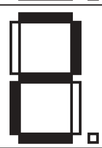
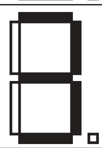
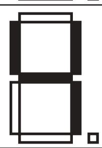
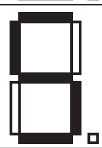
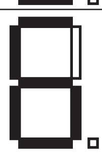
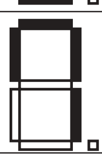
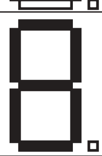
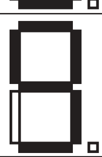
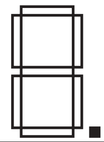
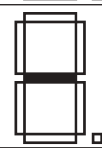


Figure 2.2 7-Segment LED Connection Diagram and Internal Wiring

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

LED Display	LED DisplayPort 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

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

Note: The first integer digit is ORed with the decimal point.

3. Principles of Operation

1. Figure 3.1 shows the principle of operation in the use of timer A and A/D conversion carried out on the AN0 pin. The A/D conversion interrupt is not used in this sample task. Instead, the completion of A/D conversion is detected in the tmra interrupt processing routine.

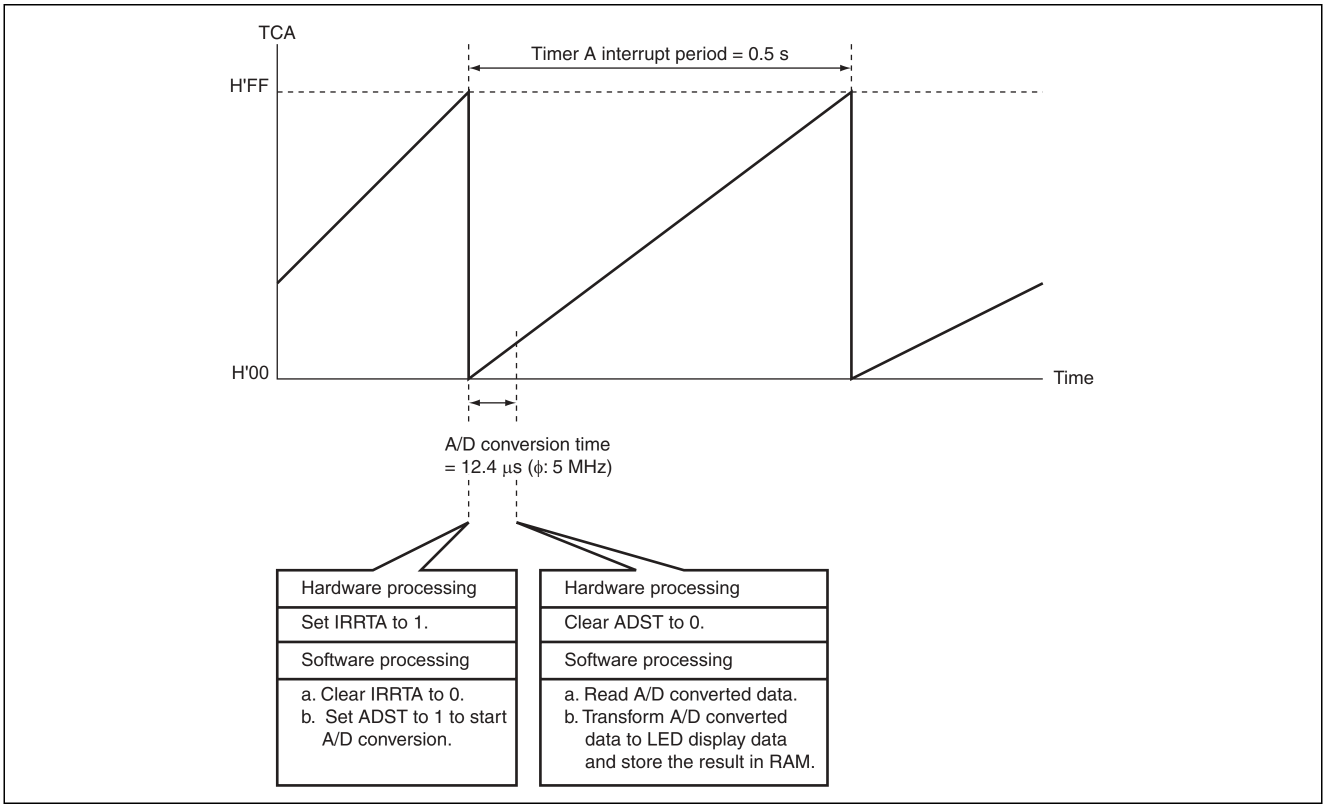


Figure 3.1 Operation Principle of A/D Conversion of AN0-Pin Signal Using Timer A

2. The principle applied in controlling the 7-segment displays is explained below. Figure 3.2 depicts the situation where "−0.50" is being displayed on LED4 to LED1. As the figure shows, the next display in sequence of LED1 to LED4 is lit up each time a timer-C overflow period elapses, creating a dynamic display on the 7-segment LEDs

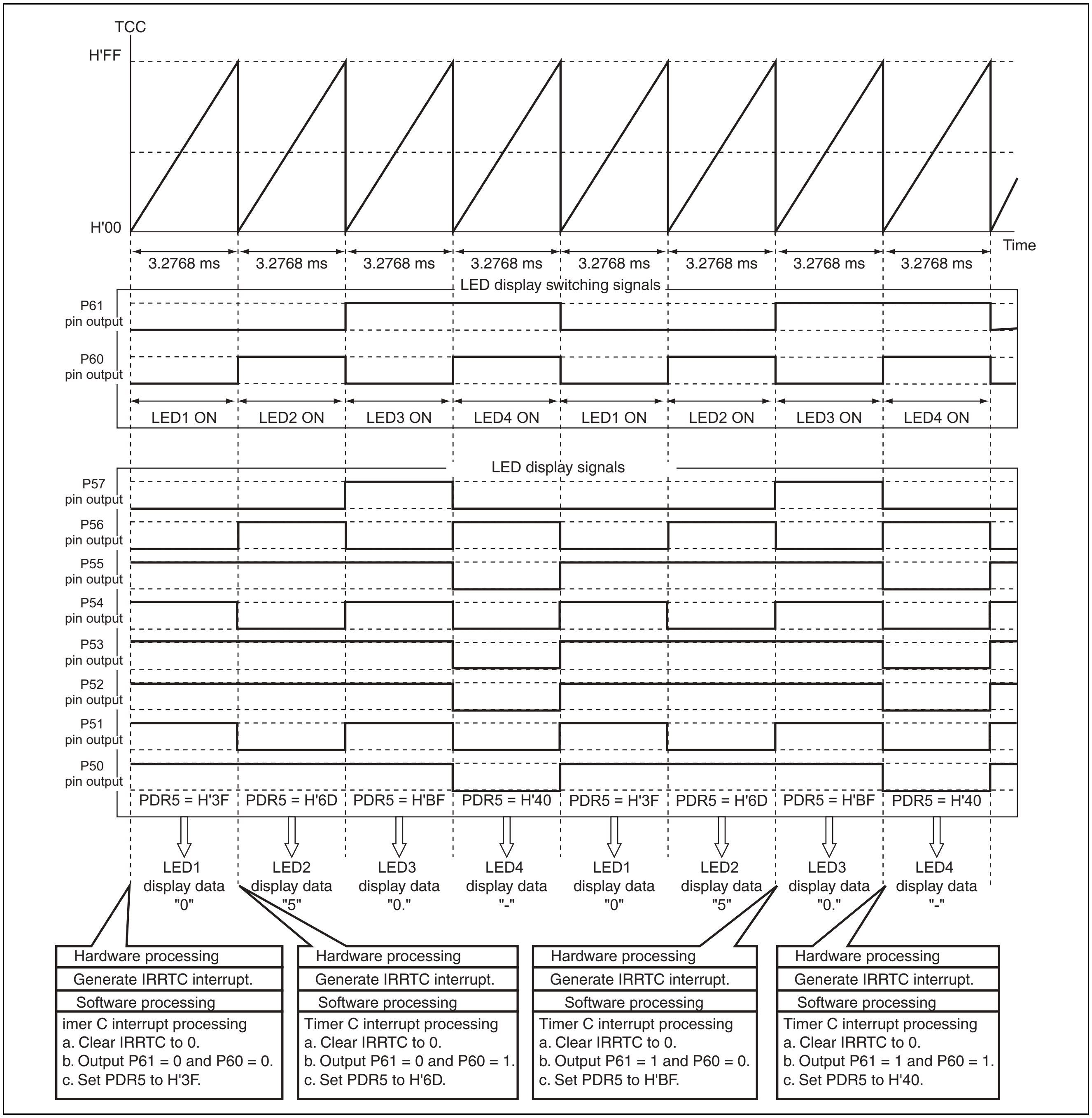


Figure 3.2 Operation Principle of 7-Segment LED Display Control

4. Description of Software

4.1 Modules

Table 4.1 describes the modules used in this sample task.

Table 4.1 Description of Modules

Module	Label	Function
Main routine	main	Makes initial settings and enables interrupts.
Timer A interrupt processing routine	tmra	Clears the interrupt flags, transforms A/D-converted data into LED-display data, and stores the result in RAM.
Timer C interrupt processing routine	tmrc	Clears the interrupt flags and controls output of LED-display data and switching of LED display.

4.2 Arguments

This sample task does not use arguments.

4.3 Internal Registers

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

Table 4.2 Description of Internal Registers

Register		Function	Address	Setting
TMA		Timer mode register A Selects prescaler and input clock.	H'FFB0	H'0C (initial setting)
	TMA3	Internal Clock Select 3 Selects timer A operating mode. When TMA3 = 1, timer A functions as the clock time base by counting the output from prescaler W.	Bit 3	1
	TMA2	Internal Clock Select 2 to 0 When TMA3 = 1, clock time base (32.768 kHz) operation is selected. When TMA2 = 1, TMA1 = 0 and TMA0 = 0, TCA is reset. When TMA2 = 0, TMA1 = 0 and TMA0 = 1, TCA overflow period is 0.5 s.	Bit 2	0/1
	TMA 1		Bit 1	0
	TMA 0		Bit 0	0/1
TMC		Timer Mode Register C Selects the automatic reloading function, controls counting upward/downward of the counter, and controls input clock.	H'FFB4	H'1B
	TMC7	Automatic Reloading Select When TMC7 = 0, the interval timer function is selected.	Bit 7	0
	TMC6	Counter Upward/Downward Control When TMC6 = 0 and TMC5 = 0, TCC is an up-counter.	Bit 6	0
	TMC5		Bit 5	0
	TMC2	Clock Select When TMC2 = 0, TMC1 = 1 and TMC0 = 1, counts on the internal clock ϕ/64.	Bit 2	0
	TMC1		Bit 1	1
	TMC0		Bit 0	1

Register		Function	Address	Setting
TLC		Timer Load Register C Sets TCC reload value.	H'FFB5	H'00
AMR		A/D Mode Register Sets A/D conversion speed, selects use of external trigger, and specifies analog input pin.	H'FFC6	H'34
	CKS	A/D Conversion Speed Setting When ϕ = 5 MHz, CKS = 0 selects 12.4 µs.	Bit 7	0
	TRGE	Trigger Enable When TRGE = 0, starting of A/D conversion in response to an external trigger input is disabled.	Bit 6	0
	CH3	Channel Select Bits 3 to 0 CH3 = 0, CH2 = 1, CH1 = 0 and CH0 = 0 are set to select AN0.	Bit 3	0
	CH2		Bit 2	1
	CH1		Bit 1	0
	CH0		Bit 0	0
ADSR		A/D Start Register Sets to start or stop A/D conversion.	H'FFC7	—
	ADSF	A/D Conversion Start/Completion Check When read: ADSF = 0 indicates that A/D conversion is complete. ADSF = 1 indicates that A/D conversion is in progress. When written: Writing ADSF = 0 forcibly terminates A/D. Writing ADSF = 1 starts A/D conversion.	Bit 7	0/1
ADRRH		A/D Result Register Stores the upper 8 bits of the results of A/D conversion.	H'FFC4	—
ADRRL		A/D Result Register Stores the lower two bits of the results of A/D conversion in bits 7 and 6.	H'FFC5	—
PUCR6		Port Pull-Up Control Register 6 Provides bit-by-bit control of the MOS pull-up for the pins of port 6 that have been set as inputs. When PUCR6 = H'00, the MOS pull-up for the P67 to P60 pins are turned off.	H'FFE3	H'00
PDR6		Port Data Register 6 General-purpose I/O port data register for port 6	H'FFD9	H'00
PCR6		Port Control Register 6 Provides bit-by-bit control of input/output selection for the pins of port 6 that have been set as general-purpose I/O pins. When PCR6 = H'FF, the pins P67 to P60 function as general-purpose output pins.	H'FFE9	H'FF

Register		Function	Address	Setting
PMR5		Port Mode Register 5 Sets the port 5 pin functions	H'FFCC	H'00
	WKP7	P57/$\overline{\text{WKP7}}$/SEG7 Pin Function Switching WKP7 = 0 selects the general-purpose I/O port function for P57.	Bit 7	0
	WKP6	P56/$\overline{\text{WKP6}}$/SEG6 Pin Function Switching WKP6 = 0 selects the general-purpose I/O port function for P56.	Bit 6	0
	WKP5	P55/$\overline{\text{WKP5}}$/ADTRG Pin Function Switching WKP5 = 0 selects the general-purpose I/O port function for P55.	Bit 5	0
	WKP4	P54/$\overline{\text{WKP4}}$ Pin Function Switching WKP4 = 0 selects the general-purpose I/O port function for P54.	Bit 4	0
	WKP3	P53/$\overline{\text{WKP3}}$ Pin Function Switching WKP3 = 0 selects the general-purpose I/O port function for P53.	Bit 3	0
	WKP2	P52/$\overline{\text{WKP2}}$ Pin Function Switching WKP2 = 0 selects the general-purpose I/O port function for P52.	Bit 2	0
	WKP1	P51/$\overline{\text{WKP1}}$ Pin Function Switching WKP1 = 0 selects the general-purpose I/O port function for P51.	Bit 1	0
	WKP0	P50/$\overline{\text{WKP0}}$ Pin Function Switching WKP0 = 0 selects the general-purpose I/O port function for P50.	Bit 0	0
PUCR5		Port Pull-Up Control Register 5 Provides bit-by-bit control of the MOS pull-up for the pins of port 5 that have been set as inputs. When PUCR5 = H'00, the MOS pull-up for the P57 to P50 pins are turned off.	H'FFE2	H'00
PDR5		Port Data Register 5 General-purpose I/O port data register for port 5	H'FFD8	H'00
PCR5		Port Control Register 5 Provides bit-by-bit control of input/output selection for the pins of port 5 that have been set as general-purpose I/O pins. When PCR5 = H'FF, the pins P57 to P50 function as general-purpose output pins.	H'FFE8	H'FF
IENR1		Interrupt Enable Register 1 Enables/disables interrupt requests.	H'FFF3	—
	IENTA	Timer A Interrupt Request Enable When IENTA = 1, timer A overflow interrupt requests are enabled.	Bit 5	1
IRR1		Interrupt Request Register 1 If an interrupt request is generated by the timer A, IRQ4, IRQ3, IRQAEC, IRQ1 or IRQ0, the corresponding flag is set to 1.	H'FFF6	—
	IRRTA	Timer A Interrupt Request Flag This is set to 1 when the timer A counter has overflowed (H'FF → H'00). This is cleared to 0 when 0 is written to.	Bit 7	0/1
IENR2		Interrupt Enable Register 2 Enables/disables interrupt requests.	H'FFF4	—
	IENTC	Timer C Interrupt Request Enable: When IENTC = 1, timer A overflow/underflow interrupt requests are enabled.	Bit 1	1

Register		Function	Address	Setting
IRR2		Interrupt Request Register 2 If an interrupt request is generated by a direct transition, A/D converter, timer G, timer FH, timer FL, timer C or asynchronous event counter, the corresponding flag is set to 1.	H'FFF7	—
	IRRTC	Timer C Interrupt Request Flag This is set to 1 when the timer C counter has overflowed (H'FF → H'00) or underflowed (H'00 → H'FF). This is cleared to 0 when 0 is written to.	Bit 7	0/1

4.4 Description of RAM

Table 4.3 describes the RAM used in this sample task.

Table 4.3 Description of RAM

Label	Function	Address	Used in
dig_0	Stores LED1 display data. (1 byte)	H'FB88	main, tmra
dig_1	Stores LED2 display data. (1 byte)	H'FB89	main, tmra
dig_2	Stores LED3 display data. (1 byte)	H'FB8A	main, tmra
dig_3	Stores LED4 display data. (1 byte)	H'FB8B	main, tmra
UCA_{OUT}	Voltage value	H'FB80	tmra
g	Acceleration value	H'FB84	tmra
SCALE	Scale factor	H'FB88	main, tmra
lednum0	LED1 display data	H'FB8C	tmra
lednum1	LED2 display data	H'FB8E	tmra
lednum2	LED3 display data	H'FB90	tmra
*ptr	Pointer to store dig_0 address	H'FB92	tmrc
cnt	8-bit counter used in switching display on LED1 to LED4. (1 byte)	H'FB98	main, tmrc

4.5 Description of Data Table

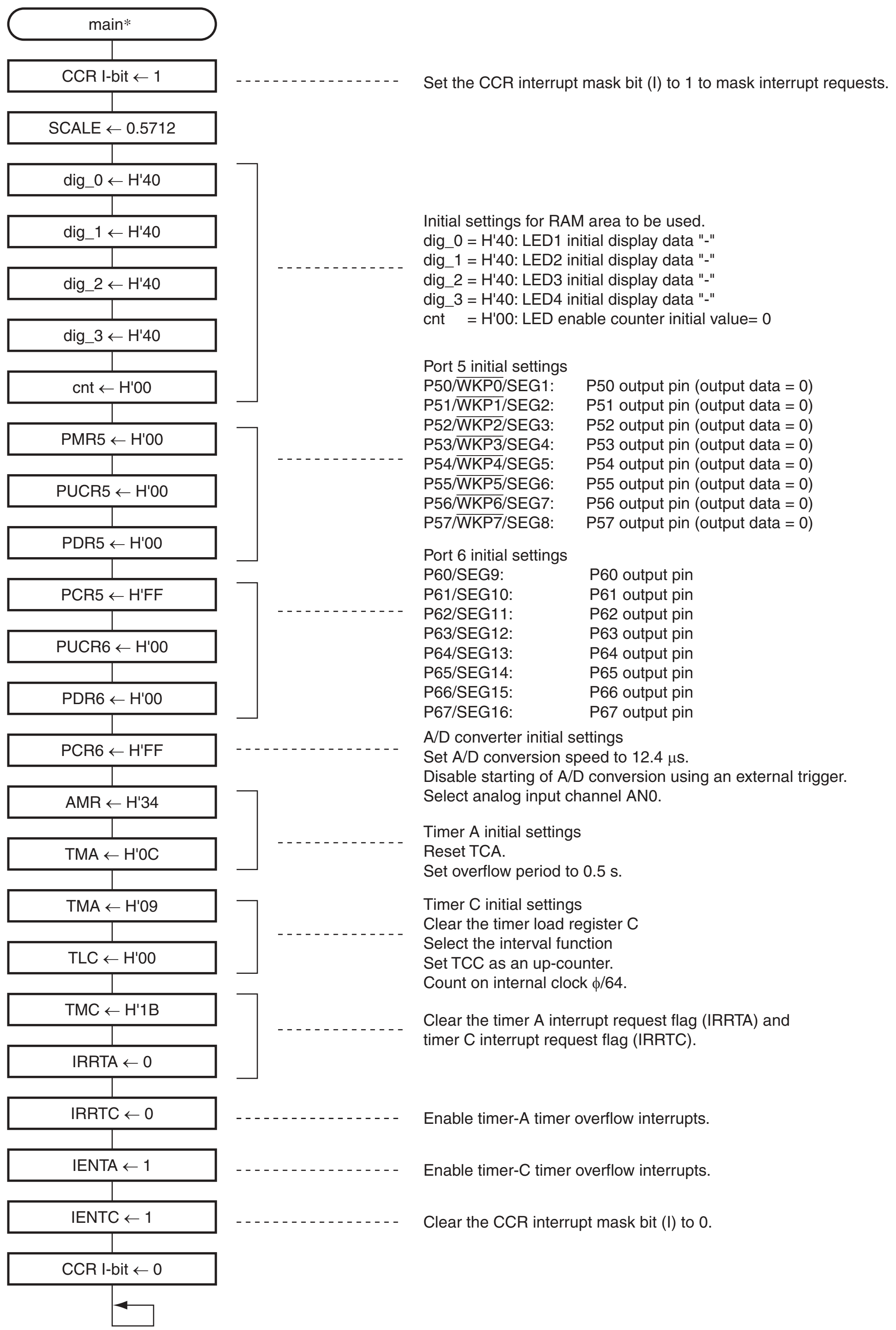
In this sample task, display data for the 7-segment LED displays are stored in the ROM as a 1-dimensional array (data table). Table 4.4 describes the table of display data (dsp_data [ ]).

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"	1 byte	H'1496
dsp_data[1]	H'06	Data output from port 5 to display "1"	1 byte	H'1497
dsp_data[2]	H'5B	Data output from port 5 to display "2"	1 byte	H'1498
dsp_data[3]	H'4F	Data output from port 5 to display "3"	1 byte	H'1499
dsp_data[4]	H'66	Data output from port 5 to display "4"	1 byte	H'149A
dsp_data[5]	H'6D	Data output from port 5 to display "5"	1 byte	H'149B
dsp_data[6]	H'7D	Data output from port 5 to display "6"	1 byte	H'149C
dsp_data[7]	H'27	Data output from port 5 to display "7"	1 byte	H'149D
dsp_data[8]	H'7F	Data output from port 5 to display "8"	1 byte	H'149E
dsp_data[9]	H'6F	Data output from port 5 to display "9"	1 byte	H'149F

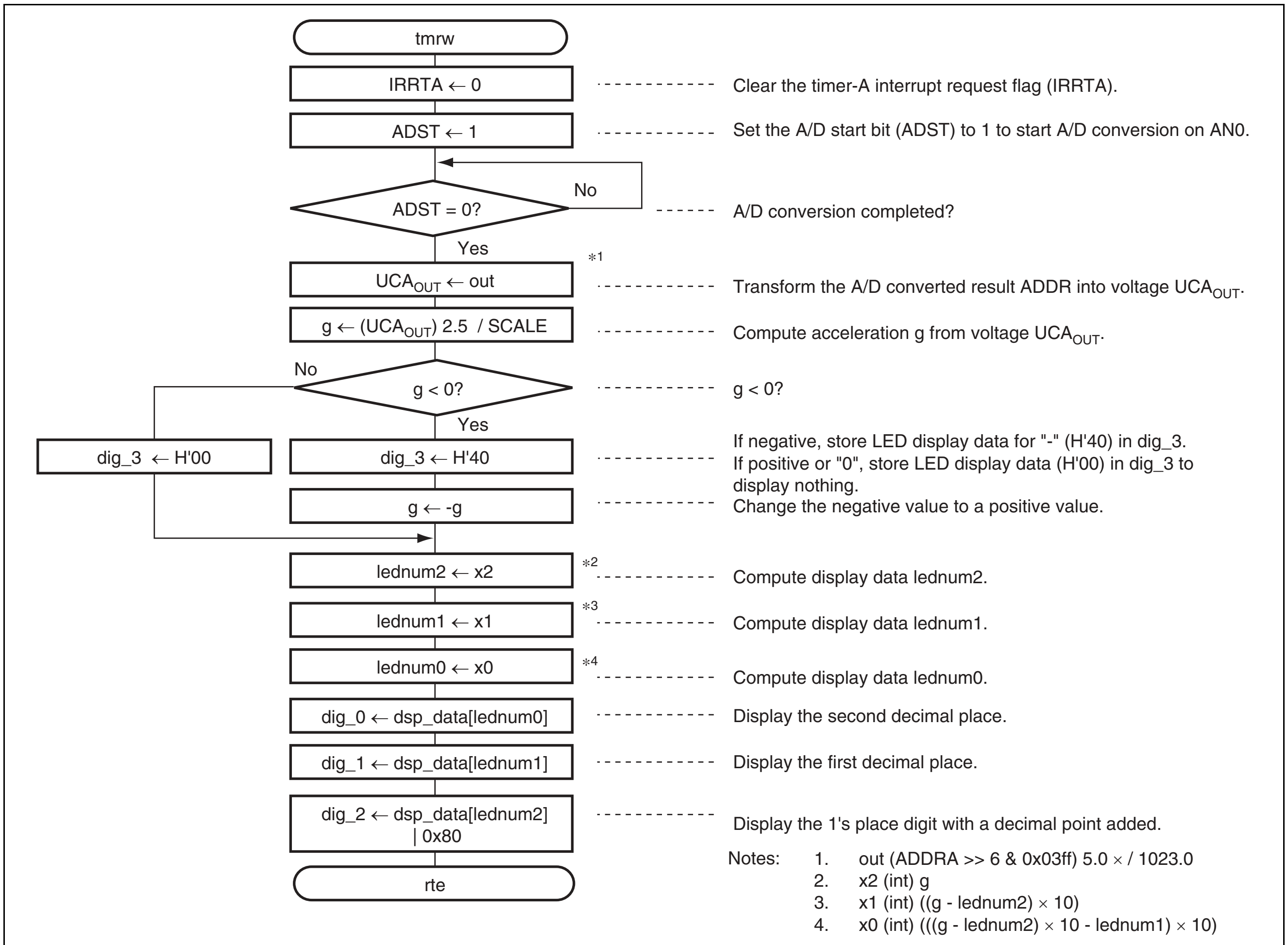
5. Flowchart

1. Main routine (main)

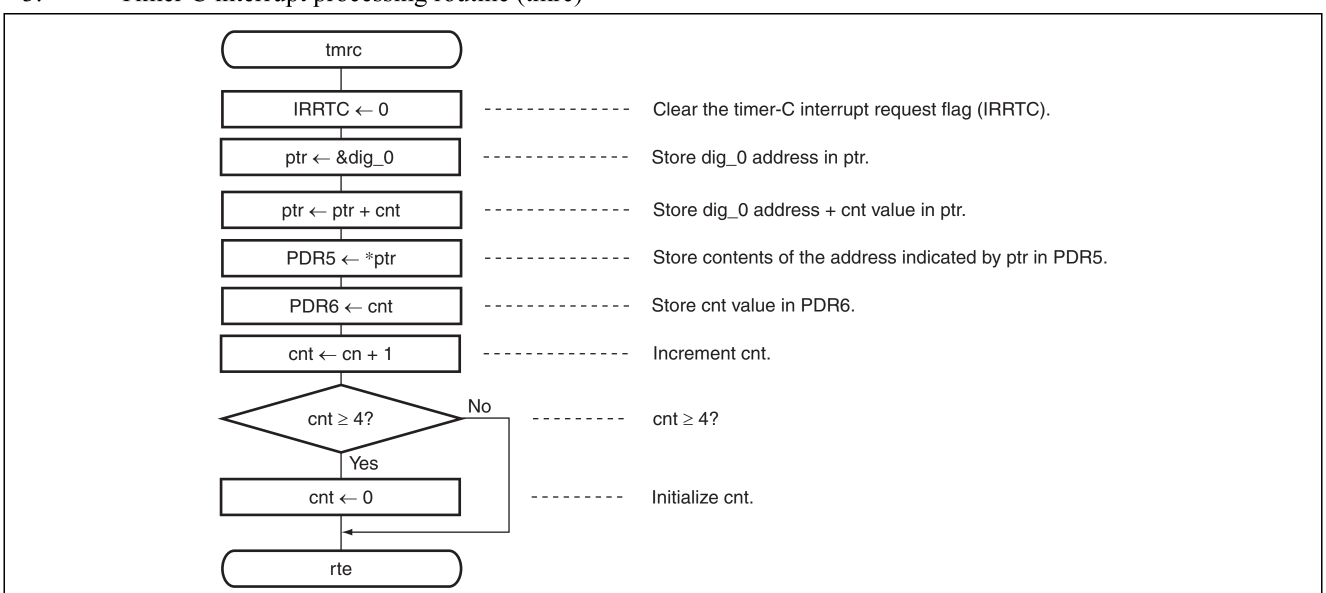


Note: * In this sample task, the stack pointer is specified by INIT.SRC (assembly language).

2. Timer A interrupt processing routine (tmra)



3. Timer C interrupt processing routine (tmrc)



6. Program Listing

INIT.SRC (Program listing)

```
    .EXPORT   _INIT
    .IMPORT   _main
 ;
    .SECTION  P,CODE
 _INIT:
    MOV.W     #H'FF80,R7
    LDC.B     #B'10000000,CCR
    JMP       @_main
 ;
    .END
```

```
/*************************************************************************************************************/
/*                                                                                                           */
/*  H8/300L Super Low Power Series                                                                           */
/*      -H8/38024 Series-                                                                                    */
/*  Application Note                                                                                         */
/*  ' Application example '                                                                                  */
/*  ' Semiconductor type acceleration sensor connection example '                                            */
/*                                                                                                           */
/*  Function                                                                                                 */
/*  : Semiconductor type acceleration sensor connection example                                              */
/*                                                                                                           */
/*  External Clock : 10MHz                                                                                   */
/*  Internal Clock : 5MHz                                                                                    */
/*  Sub Clock      : 32.768kHz                                                                               */
/*                                                                                                           */
/*************************************************************************************************************/

#include <machine.h>

/*************************************************************************************************************/
/*  Symbol Defnition                                                                                         */
/*************************************************************************************************************/
struct BIT {
    unsigned char   b7:1;       /* bit7 */
    unsigned char   b6:1;       /* bit6 */
    unsigned char   b5:1;       /* bit5 */
    unsigned char   b4:1;       /* bit4 */
    unsigned char   b3:1;       /* bit3 */
    unsigned char   b2:1;       /* bit2 */
    unsigned char   b1:1;       /* bit1 */
    unsigned char   b0:1;       /* bit0 */
};

#define     PMR5        *(volatile unsigned char *)0xFFCC       /* Port mode register 5                  */
#define     PUCR5       *(volatile unsigned char *)0xFFE2       /* 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     PUCR6       *(volatile unsigned char *)0xFFE3       /* Port pull-up control register 6       */
#define     PDR6        *(volatile unsigned char *)0xFFD9       /* Port data register 6                  */
#define     PCR6        *(volatile unsigned char *)0xFFE9       /* Port control register 6               */
```

```
#define    TMA         *(volatile unsigned char *)0xFFB0        /* Timer mode register A                */
#define    CKSTPR1     *(volatile unsigned char *)0xFFFA        /* Clock stop register 1                */

#define    TMC         *(volatile unsigned char *)0xFFB4        /* Timer mode register C                */
#define    TLC         *(volatile unsigned char *)0xFFB5        /* Timer Load register C                */


#define    ADRR        *(volatile unsigned int  *)0xFFC4        /* A/D result register (word access)    */
#define    ADRRH       *(volatile unsigned int  *)0xFFC4        /* A/D result register (byte access)    */
#define    ADRRL       *(volatile unsigned int  *)0xFFC5        /* A/D result register (byte access)    */
#define    AMR         *(volatile unsigned char *)0xFFC6        /* A/D mode register                    */
#define    ADSR        *(volatile unsigned char *)0xFFC7        /* A/D start register                   */
#define    ADSR_BIT    (*(struct BIT *)0xFFC7)
#define    ADST        ADSR_BIT.b7                              /* A/D start                            */


#define    IRR1        *(volatile unsigned char *)0xFFF6        /* Interrupt request register 1         */
#define    IRR1_BIT    (*(struct BIT *)0xFFF6)
#define    IRRTA       IRR1_BIT.b7                              /* Timer A interrupt request flag       */
#define    IENR1       *(volatile unsigned char *)0xFFF3        /* Interrupt enable register 1          */
#define    IENR1_BIT   (*(struct BIT *)0xFFF3)
#define    IENTA       IENR1_BIT.b7                             /* Timer A interrupt enable             */

#define    IRR2        *(volatile unsigned char *)0xFFF7        /* Interrupt request register 2         */
#define    IRR2_BIT    (*(struct BIT *)0xFFF7)
#define    IRRTC       IRR2_BIT.b1                              /* Timer C interrupt request flag       */
#define    IENR2       *(volatile unsigned char *)0xFFF4        /* Interrupt enable register 2          */
#define    IENR2_BIT   (*(struct BIT *)0xFFF4)
#define    IENTC       IENR2_BIT.b1                             /* Timer C interrupt enable             */

#pragma interrupt (tmra)
#pragma interrupt (tmrc)

/*****************************************************************************************************/
/*  Function Definition                                                                              */
/*****************************************************************************************************/
extern void INIT(void);                                     /* Stack pointer set                    */
void main(void);                                            /* main routine                         */
void tmra(void);                                            /* Timer A interrupt routine            */
void tmrc(void);                                            /* Timer C 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                    */
float UCAout;                                           /* Output voltage                        */
float g;                                                /* Acceleration scale result (unit: g)   */
float SCALE;                                            /* Scale factor for 3.3V                 */
unsigned char *ptr;                                     /* Pointer set                           */

/******************************************************************************************************/
/*  Vector Address                                                                                     */
/******************************************************************************************************/
#pragma section V1                                      /* Vector section set                    */
void (*const VEC_TBL1[])(void) = {
    INIT                                                /* 0x0000 Reset vector                   */
};
#pragma section V2                                      /* Vector section set                    */
void (*const VEC_TBL2[])(void) = {
    tmra                                                /* 0x0016 Timer A interrupt vector       */
};
#pragma section V3                                      /* Vector section set                    */
void (*const VEC_TBL3[])(void) = {
    tmrc                                                /* 0x001A Timer C interrupt vector       */
};
#pragma section                                         /* P                                     */

/******************************************************************************************************/
/*  Main Program                                                                                       */
/******************************************************************************************************/
void main(void)
{
    set_imask_ccr(1);                                   /* CCR I-bit = 1                         */
    SCALE = 0.5712;                                     /* Scale factor for 3.3V                 */

    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              */

    PMR5  = 0x00;                                       /* Port 5 initialize                     */
    PUCR5 = 0x00;
    PDR5  = 0x00;
    PCR5  = 0xff;

    PUCR6 = 0x00;                                       /* Port 6 initialize                     */
    PDR6  = 0x00;
    PCR6  = 0xFF;

    AMR   = 0x34;                                       /* A/D converter initialize (AN0)        */

    TMA   = 0x0c;                                       /* Clear Timer Counter A to 0            */
    TMA   = 0x09;                                       /* Timer A initialize                    */
    TLC   = 0x00;                                       /* Clear Timer Load register C to 0      */
    TMC   = 0x1b;                                       /* Timer C initialize                    */
```

```
    IRRTA = 0;                                                  /* Clear IRRTA to 0                  */
    IRRTC = 0;                                                  /* Clear IRRTC to 0                  */
    IENTA = 1;                                                  /* Timer A interrupt enable          */
    IENTC = 1;                                                  /* Timer C interrupt enable          */

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

    while(1);
}


/*************************************************************************************************************/
/*  Timer A Interrupt                                                                                        */
/*************************************************************************************************************/
void tmra(void)
{
    IRRTA = 0;                                                  /* Clear IRRTA to 0                  */
    ADST  = 1;                                                  /* A/D converter start               */
    while(ADST == 1);                                           /* A/D converter end ?               */
    UCAout = (ADRR >> 6 & 0x03ff) * 3.3 / 1023.0;               /* Output voltage                    */
    g = (UCAout - 1.65) / 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 C Interrupt                                                                                            */
/*****************************************************************************************************************/
void tmrc(void)
{
    IRRTC = 0;                                                  /* Clear IRRTC to 0                         */

    ptr = &dig_0;                                               /* LED display data store address set       */
    ptr += cnt;                                                 /* LED display data read                    */
    PDR5 = *ptr;                                                /* LED display data output                  */
    PDR6 = cnt;                                                 /* LED enable data output                   */

    cnt++;                                                      /* "cnt" increment                          */
    if (cnt >= 4){                                              /* 4 times end ?                            */
        cnt = 0;                                                /* "cnt" initialize                         */
    }
}
```

Link address specifications

Section Name	Address
CV1	H'0000
CV2	H'0016
CV3	H'001A
P	H'0100
B	H'FB80

Revision Record

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