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

Creating a DTMF Generator Using PWM

Introduction

This document describes how to create a Dual Tone Multifrequency (DTMF) generator using a D/A converter with Pulse Width Modulation (PWM) to place calls from a general-purpose telephone.

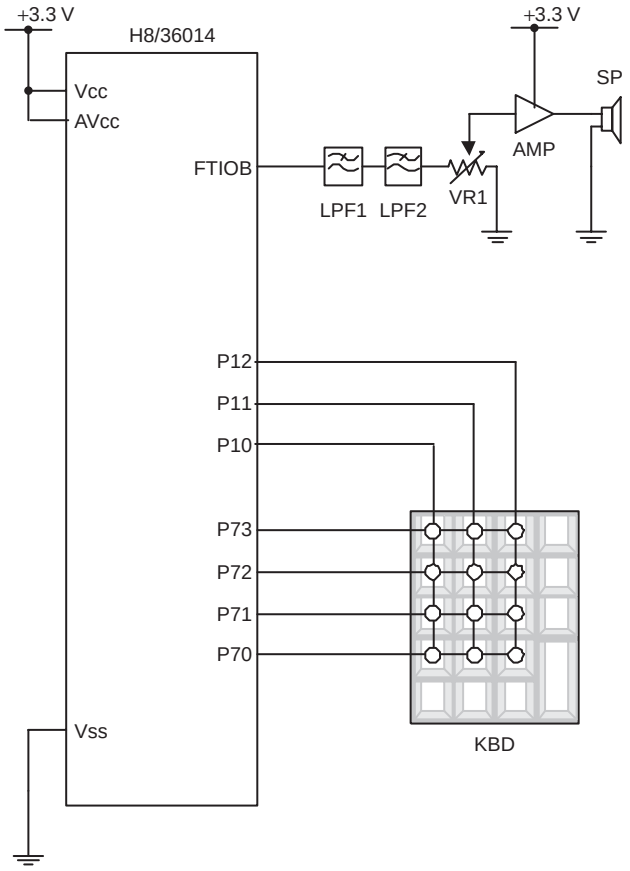
Target Device

H8/36014

Contents

1. Specifications

- Figure 1 shows the hardware configuration of the Dual Tone Multifrequency (DTMF) generator using a D/A converter and Pulse Width Modulation (PWM) technology.
- In this task, pressing a key on the numeric keypad synthesizes two types of sine waves and outputs the sum to a speaker.
- In this task, a numeric keypad for personal computers is used. This keypad normally outputs one-bit serial signals. However, it is modified to perform parallel input and output.



Legend:

LPF1	: VCVS secondary low pass filter (f_0 = 3 kHz)
LPF2	: CR primary low pass filter (f_0 = 81 kHz)
VR1	: Volume (reference: 100 Ω)
AMP	: Power amplifier (NJM2073 manufactured by Shin-Nihon Musen)
SP	: Speaker
KBD	: Keypad

Figure 1 Hardware Configuration

- D/A conversion using PWM
  a. By allowing PWM waves to pass through a low pass filter, DC voltage is obtained.
  b. The DC voltage level is determined by the duty ratio (ratio of a high-level period to the cycle of a PWM wave) of PWM waves.

     Vout = V × H/T (V: amplitude, H: high-level period, T: cycle)

  c. In this example, the frequency of PWM waves is set to 31.25 kHz. The cutoff frequency of the low pass filter is set to 3 kHz, which is sufficiently lower than the frequency of the PWM clock signals and a VCVS active low pass filter is used.
  d. Since the VCVS active low pass filter uses an OP amplifier, it does not function adequately in the high-frequency area, allowing the frequencies higher than a few MHz to escape. Therefore, a CR filter is also used to filter out high frequencies.
- The DTMF generator functions as follows.
  a. Among the seven ports connected to the keypad, the DTMF generator sets P70 to P73 to the H level sequentially.
  b. The DTMF generator monitors the status of the remaining three ports (P10 to P12) and detects the “H” level on the port to determine the pressed key.
  c. To output a DTMF tone for the pressed key, the DTMF generator uses PWM to perform D/A conversion.
  d. At D/A conversion, from the output port (FTIOB) of the hardware timer W, the DTMF generator first outputs PWM waveforms which are modulated to obtain the desired peak value for the amplifier input. Since a frequency sufficiently lower than the frequency of the PWM clock signals is selected as the cutoff frequency of the low pass filter, the square waves of the high frequency is smoothed.
  e. Next, the DTMF generator changes the duty ratio according to the frequency of the DTMF tone. In this case, the DTMF tone generated by the low frequency part of the PWM is output to the speaker.
  f. Figure 2 shows the available DTMF frequencies. For example, when you press “1” on the keypad, the sum of 697 Hz and 1,209 Hz is output through the speaker. Use the volume control to adjust the volume.

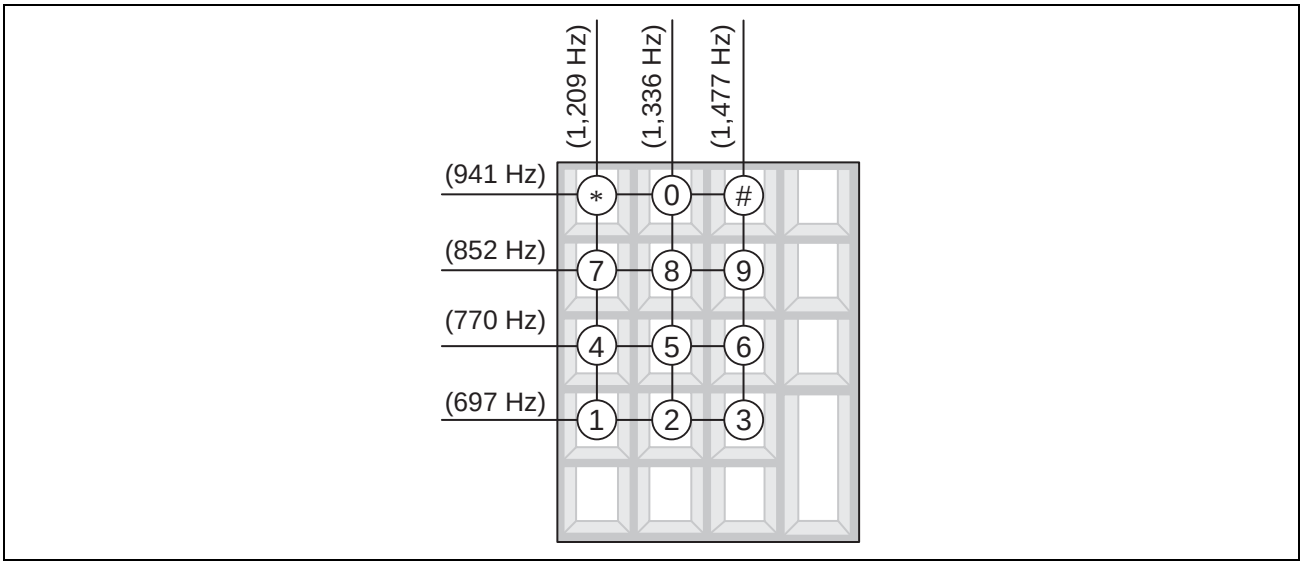


Figure 2 DTMF Frequencies

- Checking the operation
  a. The operation of the DTMF generator was checked using a general-purpose telephone.
  b. To check, the DTMF generator and the handset were placed side by side about 3 cm apart, and the generated tone was used to dial a number to determine whether the telephone could place a call.
  c. The telephone recognized all numbers from 0 to 9 and calls were correctly placed.
- Expanding keys
  a. In this task, seven signal lines are used to enable 12 keys (four rows x three columns) on a general-purpose telephone, which are 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, *, #.
  b. Figure 3 shows the extended keypad layout. To add A, B, C, and D keys, add a 1633-Hz signal line to the columns of the keypad.

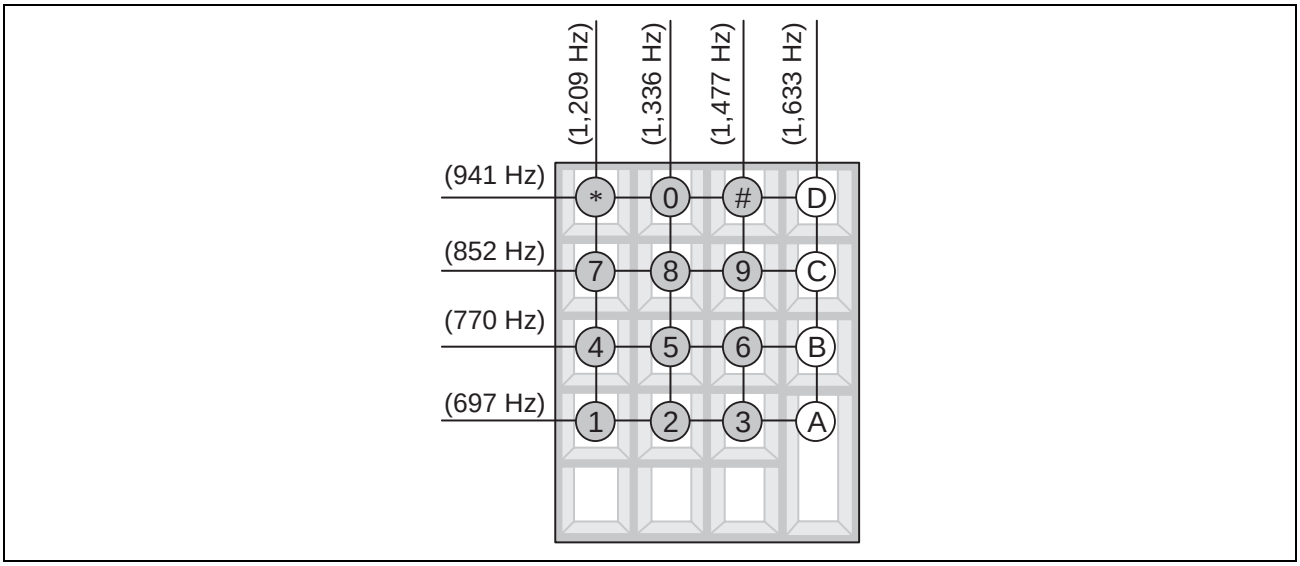


Figure 3 Extended Keypad Layout

2. Description of Functions

- Figure 4 shows the block diagram of the functions used in the H8/36014 in this task and Table 1 shows the details about each function.

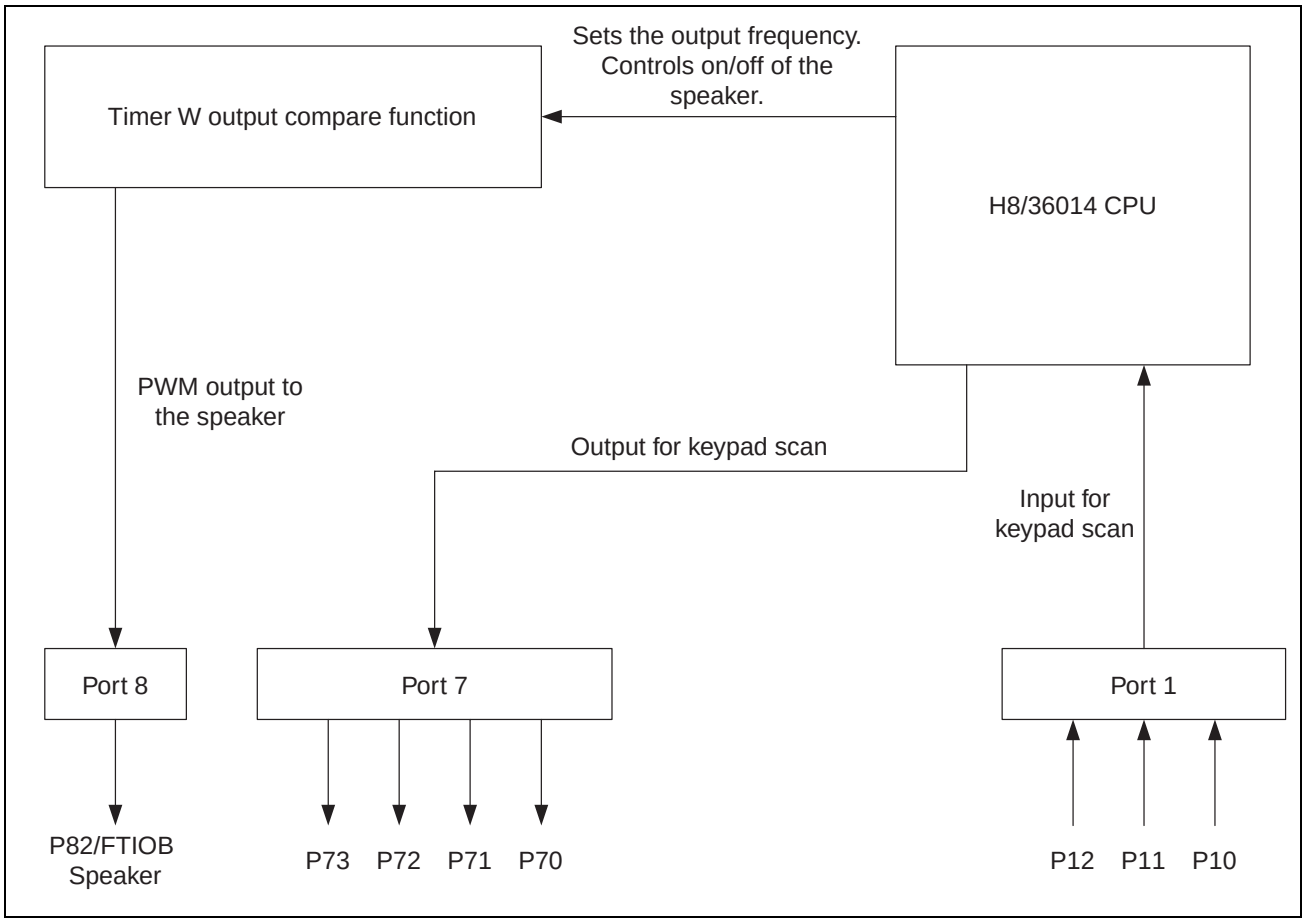


Figure 4 Block Diagram of the Functions Used

Table 1 Details about the Functions

Function	Description
Timer W	Employs the PWM function to output PWM waves. You can add a desired value to general-purpose register B (GRB) to change the output frequency.
Port 1	The P10 to P12 input pins of port 1 are used to detect keypad input.
Port 7	The P70 to P73 output pins of port 7 are sequentially set to the H level.

3. Description of Operation

1. Figure 5 describes the operation for outputting DTMF tones using timer W. As shown in Figure 5, the value of GRB is changed at every compare match cycle of timer W to output DTMF tones.

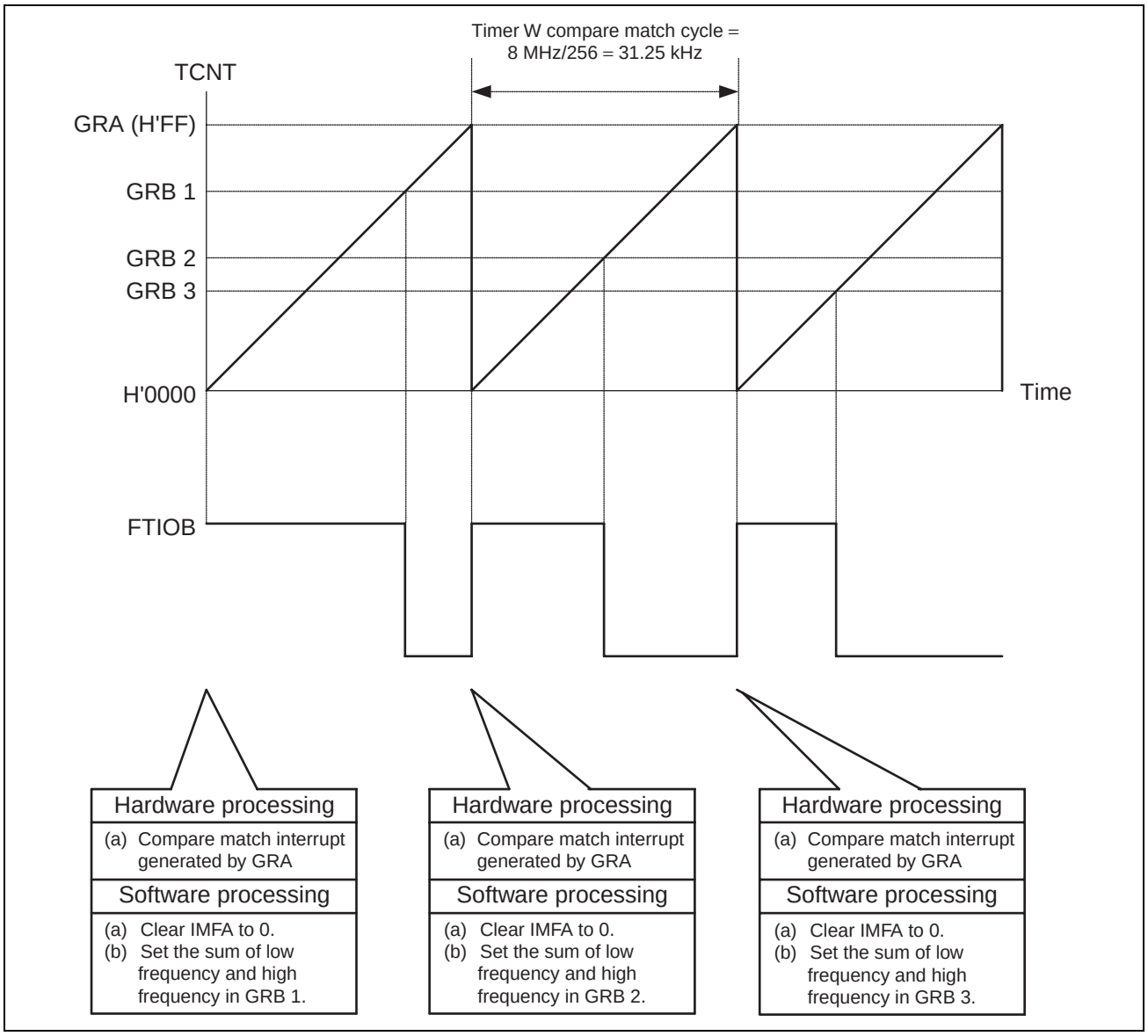


Figure 5 Operation of Outputting DTMF Tones Using Timer W

2. The numeric keypad used in this task has only 12 keys (0 to 9, *, #). If you need A, B, C, and D, change input_key(void) and the push button frequency table (PB) in the ROM definition. Since the changes you may want to add to the push button frequency table are commented out using //, remove // to use them.

4. Description of Software

4.1 About the modules

Table 2 describes the modules used in this task.

Table 2 About the Modules

Module	Label	Description
Main routine	main	Makes the initial settings and outputs frequencies.
Keyboard input processing routine	input_key	Checks the input from the numeric keypad and returns the applicable key code. When no key is pressed: - 1 When a key is pressed: 0 to 11
Timer W interrupt routine	tmrw	Updates the index of the sign table. Updates the PWM output.

4.2 About the arguments

No arguments are used in this task.

4.3 About the registers

Table 3 lists the registers used in this task.

Table 3 About the Registers

Register		Description	Address	Set value
TMRW		Timer mode register W Used to select the function of a general-purpose register and the output mode of the timer.	H'FF80	H'81
	CTS	Count start bit When CTS = 1, TCNT starts counting. When CTS = 0, TCNT stops counting.	Bit 7	1
	PMWB	When PMWB = 1, the output mode of the FTIOB pin is set to PWM output.	Bit 0	1
TCRW		Timer control register W Used to select the clock signal for the count register. Used to set the condition to clear the count register and select the output level of the timer.	H'FF81	H'82
	CCLR	Count clear bit When CCLR = 1, TCNT is cleared by compare match flag A.	Bit 7	1
	CKS2	Clock signal selection bits When CKS2 = 0, CKS1 = 0 and CKS0 = 0, system clock signals are used as the input clock signals for TCNT.	Bit 6	0
	CKS1		Bit 5	0
	CKS0		Bit 4	0
	TCB	Timer output level set B When TCB = 1, the output value of the FTIOB pin is set to 1 before the first compare match B is generated.	Bit 1	1

Register		Description	Address	Set value
TIERW		Timer interrupt enable register W Controls the interrupt requests of timer W.	H'FF82	H'00 (initial setting)
	IMIEA	Input capture/compare match interrupt enable A When IMIEA = 1, interrupt requests (IMIA) generated by IMFA of TSRW are enabled.	Bit 0	1
TSRW		Indicates the interrupt request status.	H'FF83	H'00
	IMFB	Input capture/compare match flag B	Bit 1	0
	IMFA	Input capture/compare match flag A	Bit 0	0
TCNT		Count register 16-bit up counter using every eighth system clock cycle for input clock signals.	H'FF86	H'00
GRA		General-purpose register A	H'FF88	H'FF
GRB		General-purpose register B	H'FF8A	H'00
PCR1		Port control register 1 Used to select the I/O of the pins used as general-purpose I/O pins of port 1 bit by bit. When PCR1 = H'00 The P10 to P12 pins function as general-purpose input pins.	H'FFE4	H'00
PDR1		Port data register 2 General-purpose I/O port data register for port 1	H'FFD4	H'00
PCR7		Port control register 7 Used to select the I/O of the pins used as general-purpose I/O pins of port 7 bit by bit. When PCR7 = H'0F The P70 to P73 pins function as general-purpose output pins.	H'FFEA	H'0F
PDR7		Port data register 7 General-purpose I/O port data register for port 7	H'FFDA	H'00 (initial setting)

4.4 About RAM

Table 4 shows how RAM is used in this task.

Table 4 Description about RAM

Label	Description	Address	Used by:
inc1	Stores the increment of the sine table for a low frequency (two byte).	H'FB80	main, tmrw
Inc2	Stores the increment of the sine table for a high frequency (two byte).	H'FB82	main, tmrw
pos1	Stores the acquisition position of the sine table for a low frequency (two bytes).	H'FB84	main, tmrw
pos2	Stores the acquisition position of the sine table for a high frequency (two bytes).	H'FB86	main, tmrw
key_data	Stores input key data (two bytes).	H'FB88	main
old_key	Stores the previous key data (two bytes).	H'FB8A	main
i	Stores the loop counter for key scan (two bytes).	H'FB8C	input_key
j	Stores the loop counter for key scan (two bytes).	H'FB8E	input_key
key	Stores the key data for key scan (two bytes).	H'FB90	input_key

4.5 About ROM

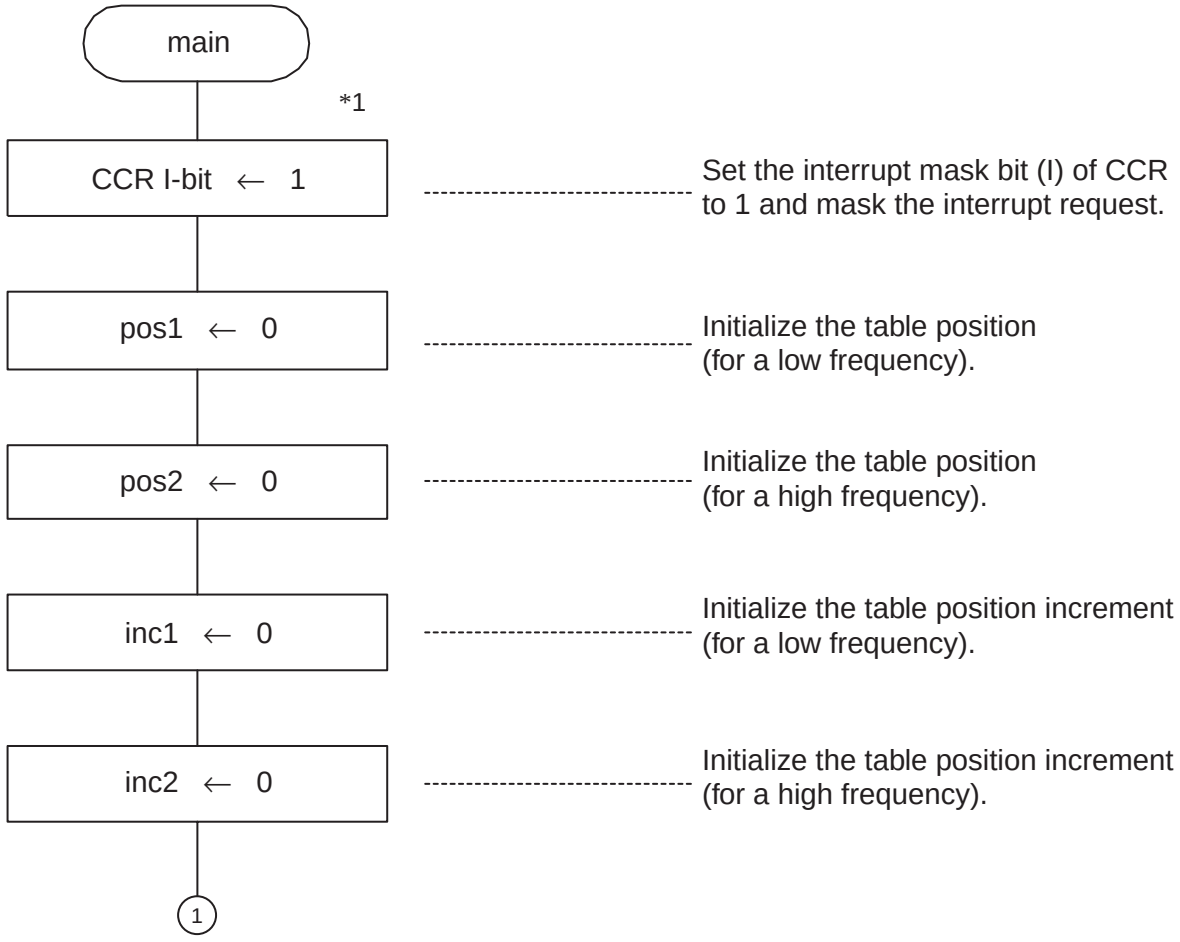
Table 5 shows how ROM is used in this task.

Table 5 Description about ROM

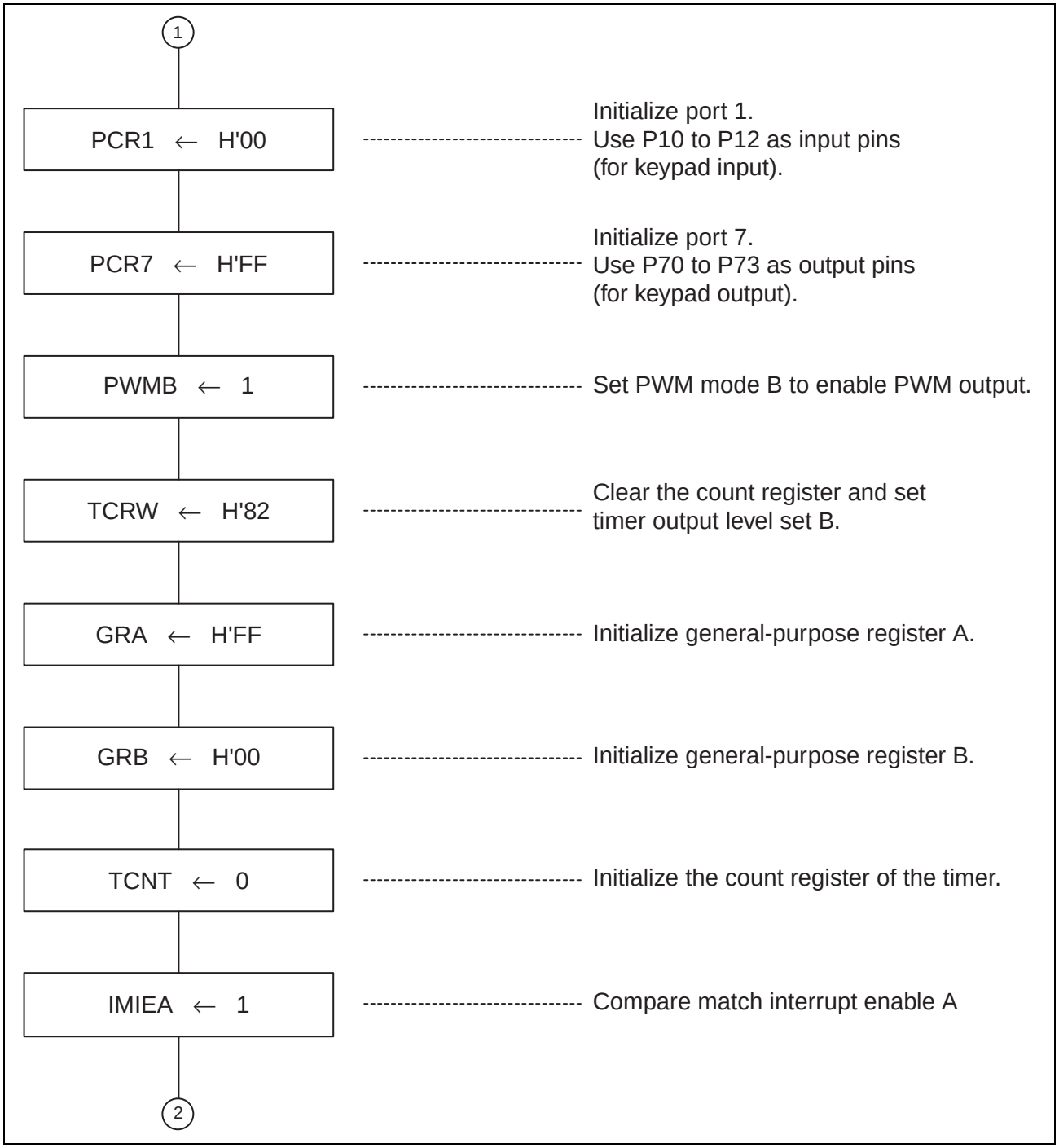
Label	Description	Address	Used by:
PB	Stores the increment of the sine table for the pressed key (48 bytes).	H'02D0	main, tmrw
sine	Stores the sine table (256 bytes).	H'0310	main, tmrw

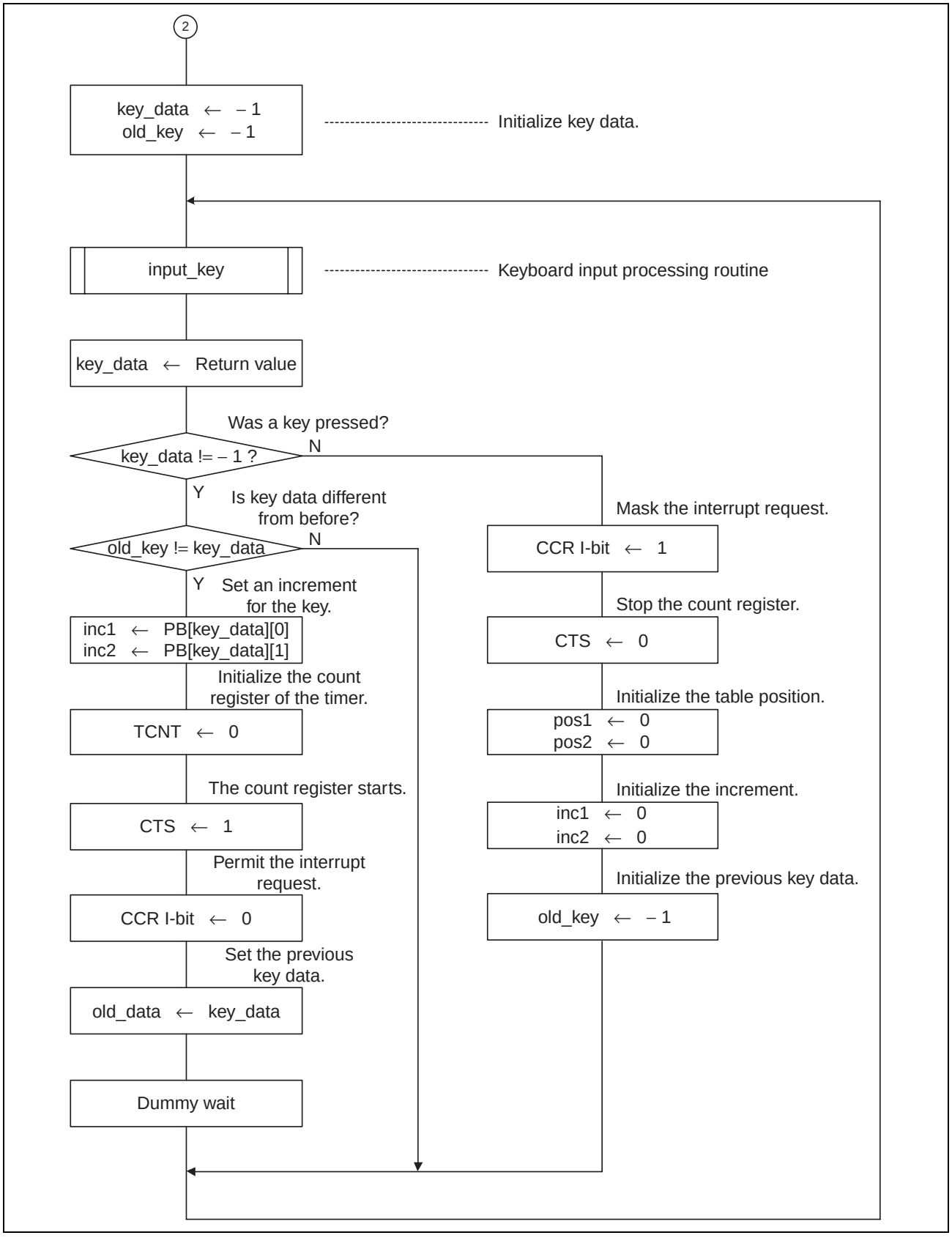
5. Flowchart

5.1 Main routine (main)

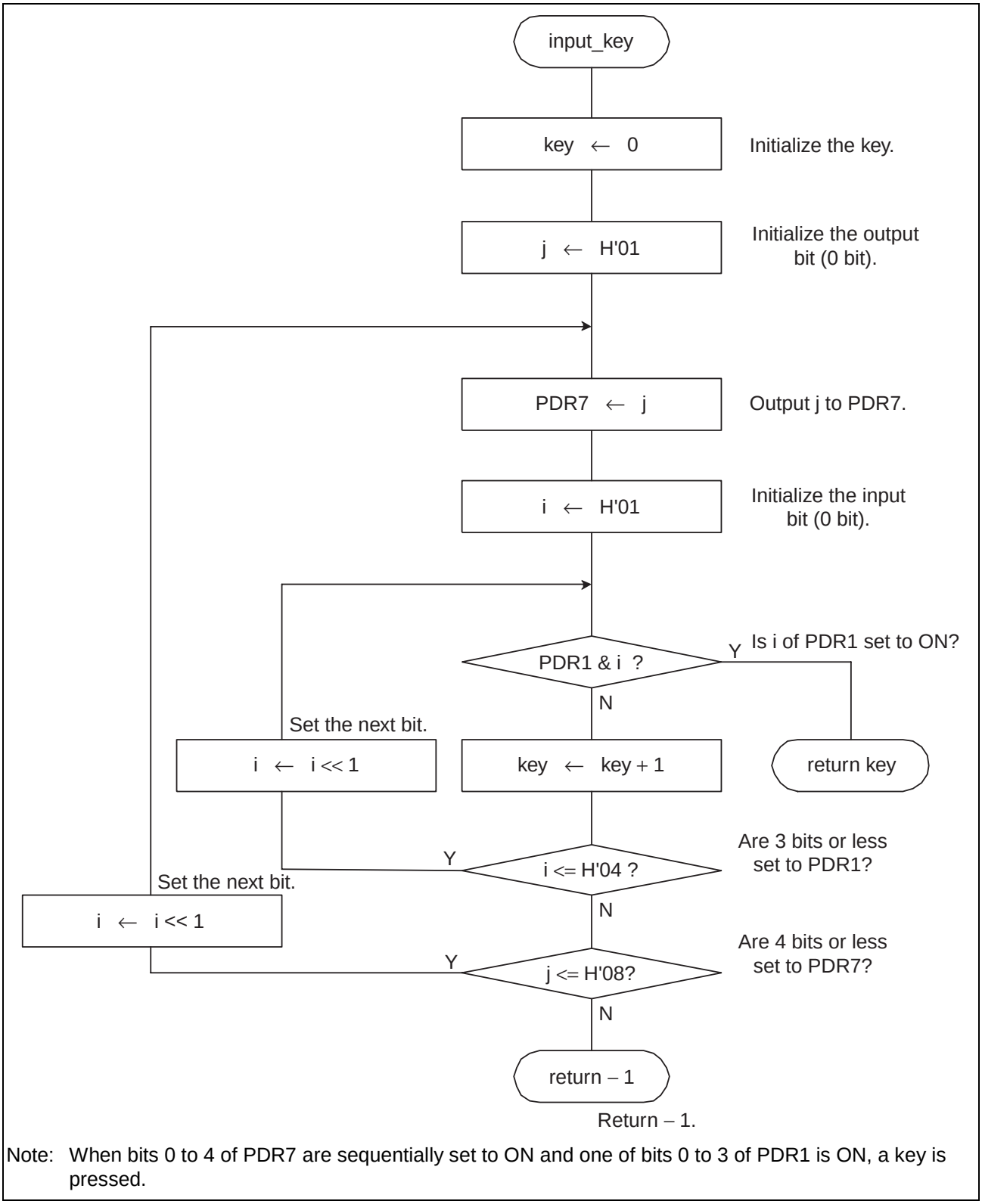


*1 The stack pointer is set using INIT.SRC (assembly language) in this task.


1
PCR1 ← H'00
Initialize port 1.
Use P10 to P12 as input pins
(for keypad input).
PCR7 ← H'FF
Initialize port 7.
Use P70 to P73 as output pins
(for keypad output).
PWMB ← 1
Set PWM mode B to enable PWM output.
TCRW ← H'82
Clear the count register and set
timer output level set B.
GRA ← H'FF
Initialize general-purpose register A.
GRB ← H'00
Initialize general-purpose register B.
TCNT ← 0
Initialize the count register of the timer.
IMIEA ← 1
Compare match interrupt enable A
2

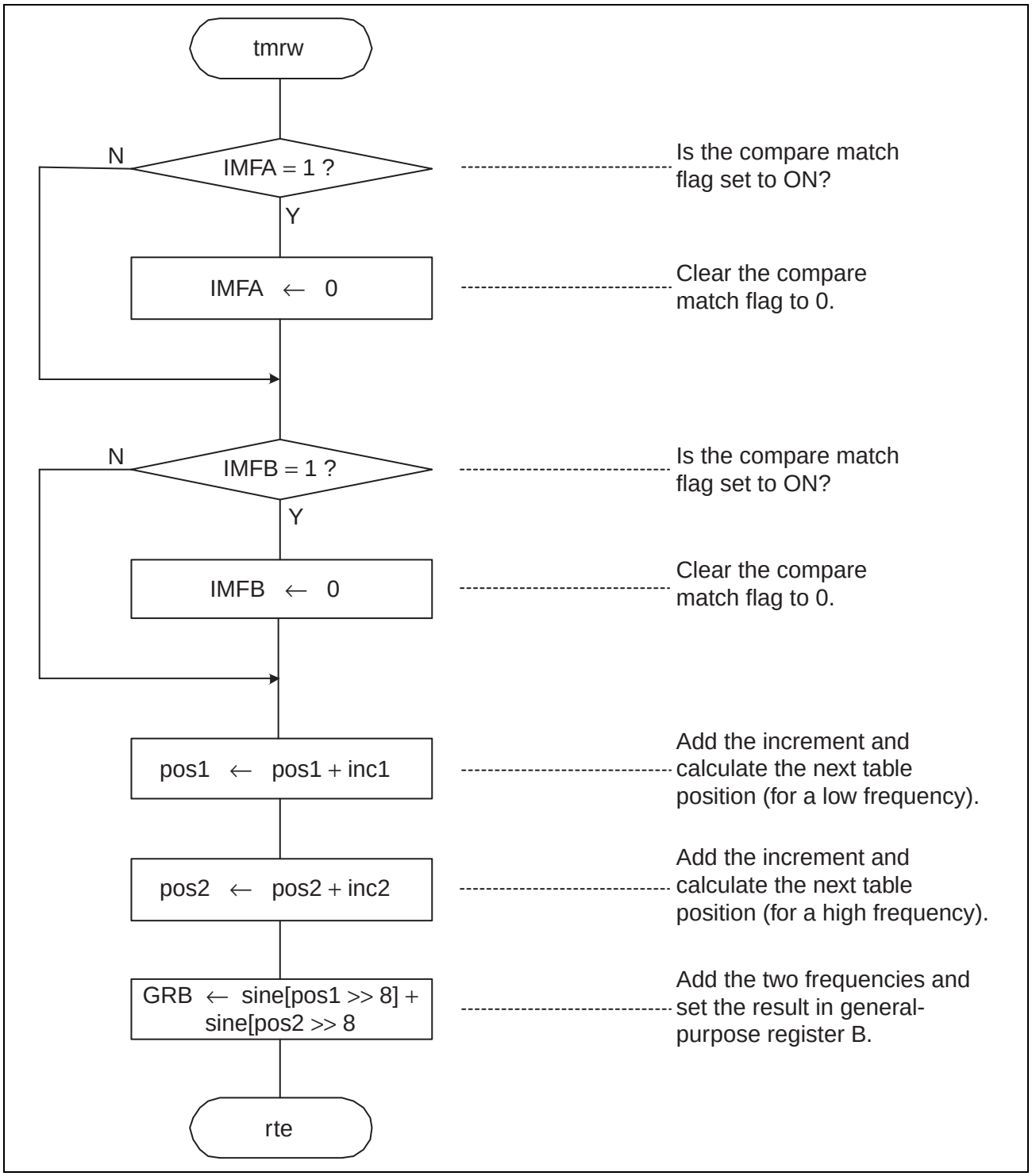

2
key_data ← – 1
old_key ← – 1
Initialize key data.
input_key
Keyboard input processing routine
key_data ← Return value
Was a key pressed?
key_data != – 1 ?
N
Y
Is key data different from before?
old_key != key_data
N
Y
Set an increment for the key.
inc1 ← PB[key_data][0]
inc2 ← PB[key_data][1]
Initialize the count register of the timer.
TCNT ← 0
The count register starts.
CTS ← 1
Permit the interrupt request.
CCR I-bit ← 0
Set the previous key data.
old_data ← key_data
Dummy wait
Mask the interrupt request.
CCR I-bit ← 1
Stop the count register.
CTS ← 0
Initialize the table position.
pos1 ← 0
pos2 ← 0
Initialize the increment.
inc1 ← 0
inc2 ← 0
Initialize the previous key data.
old_key ← – 1

5.2 Keyboard input processing routine (input_key)



Note: When bits 0 to 4 of PDR7 are sequentially set to ON and one of bits 0 to 3 of PDR1 is ON, a key is pressed.

5.3 Timer W interrupt routine (tmrw)



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
```

```
/* H8/300H tiny Series -H8/36014- Application note */
/* Application */
/* DTMF output example */

#include <machine.h>

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

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

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

#define TMRW   *(volatile unsigned char *)0xFF80          /* Timer mode register W */
#define TMRW_BIT   (*(struct BIT *)0xFF80)                /* Timer mode register W */
#define CTSTMRW_BIT.b7                                    /* Count register start */
#define PWMB   TMRW_BIT.b0                                /* PWM mode B */
#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 IMIEA  TIERW_BIT.b0               /*Input-capture/compare-match interrupt enable A */
#define TSRW   *(volatile unsigned char *)0xFF83          /* Timer status register W */
```

```
#define TSRW_BIT  (*(struct BIT *)0xFF83)                /* Timer status register W */
#define IMFA   TSRW_BIT.b0                               /* Input-capture/compare-match flag A */
#define IMFB   TSRW_BIT.b1                               /* Input-capture/compare-match flag B */
#define TIOR0  *(volatile unsigned char *)0xFF84         /* Timer I/O control register 0 */
#define GRA*(volatile unsigned short *)0xFF88            /* General-purpose register A */
#define GRB*(volatile unsigned short *)0xFF8A            /* General-purpose register B */
#define TCNT   *(volatile unsigned short *)0xFF86        /* Timer count register */

// 3.3 V 8 MHz 8000000 / 256 = 31250
#define FREQ697     697L * 256 * 256 / 31250                    /*  697 Hz */
#define FREQ770     770L * 256 * 256 / 31250                    /*  770 Hz */
#define FREQ852     852L * 256 * 256 / 31250                    /*  852 Hz */
#define FREQ941     941L * 256 * 256 / 31250                    /*  941 Hz */
#define FREQ1209   1209L * 256 * 256 / 31250                    /* 1209 Hz */
#define FREQ1336   1336L * 256 * 256 / 31250                    /* 1336 Hz */
#define FREQ1477   1477L * 256 * 256 / 31250                    /* 1477 Hz */
#define FREQ1633   1633L * 256 * 256 / 31250                    /* 1633 Hz */

#pragma interrupt (tmrw)

/* Function definition */
extern void INIT(void);                                  /* Set stack pointer */
void main(void);                                         /* Main routine */
int input_key(void);                                     /* Input key */
void tmrw(void);                                         /* Timer W interrupt routine */

/* ROM definition */
const unsigned short PB[16][2] = {                       /* Push button - freq. table */
    FREQ697, FREQ1209,                                   /* 1 */
    FREQ697, FREQ1336,                                   /* 2 */
    FREQ697, FREQ1477,                                   /* 3 */
    FREQ770, FREQ1209,                                   /* 4 */
    FREQ770, FREQ1336,                                   /* 5 */
    FREQ770, FREQ1477,                                   /* 6 */
    FREQ852, FREQ1209,                                   /* 7 */
    FREQ852, FREQ1336,                                   /* 8 */
    FREQ852, FREQ1477,                                   /* 9 */
    FREQ941, FREQ1209,                                   /* * */
    FREQ941, FREQ1336,                                   /* 0 */
    FREQ941, FREQ1477,                                   /* # */
//  FREQ697, FREQ1633,                                   /* A */
//  FREQ770, FREQ1633,                                   /* B */
//  FREQ852, FREQ1633,                                   /* C */
//  FREQ941, FREQ1633,                                   /* D */
};

const unsigned char sine[256] = {                        /* Sine table  */
    0x40, 0x42, 0x43, 0x45, 0x46, 0x48, 0x49, 0x4B,
    0x4C, 0x4E, 0x4F, 0x51, 0x52, 0x54, 0x55, 0x57,
    0x58, 0x59, 0x5B, 0x5C, 0x5E, 0x5F, 0x60, 0x62,
    0x63, 0x64, 0x65, 0x66, 0x68, 0x69, 0x6A, 0x6B,
    0x6C, 0x6D, 0x6E, 0x6F, 0x70, 0x71, 0x72, 0x73,
    0x74, 0x75, 0x75, 0x76, 0x77, 0x77, 0x78, 0x79,
    0x79, 0x7A, 0x7A, 0x7B, 0x7B, 0x7C, 0x7C, 0x7C,
```

```
    0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7E,
    0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7C,
    0x7C, 0x7C, 0x7B, 0x7B, 0x7A, 0x7A, 0x79, 0x79,
    0x78, 0x77, 0x77, 0x76, 0x75, 0x75, 0x74, 0x73,
    0x72, 0x71, 0x70, 0x6F, 0x6E, 0x6D, 0x6C, 0x6B,
    0x6A, 0x69, 0x68, 0x66, 0x65, 0x64, 0x63, 0x62,
    0x60, 0x5F, 0x5E, 0x5C, 0x5B, 0x59, 0x58, 0x57,
    0x55, 0x54, 0x52, 0x51, 0x4F, 0x4E, 0x4C, 0x4B,
    0x49, 0x48, 0x46, 0x45, 0x43, 0x42, 0x40, 0x3F,
    0x3D, 0x3B, 0x3A, 0x38, 0x37, 0x35, 0x34, 0x32,
    0x31, 0x2F, 0x2E, 0x2C, 0x2B, 0x29, 0x28, 0x26,
    0x25, 0x24, 0x22, 0x21, 0x1F, 0x1E, 0x1D, 0x1B,
    0x1A, 0x19, 0x18, 0x17, 0x15, 0x14, 0x13, 0x12,
    0x11, 0x10, 0x0F, 0x0E, 0x0D, 0x0C, 0x0B, 0x0A,
    0x09, 0x08, 0x08, 0x07, 0x06, 0x06, 0x05, 0x04,
    0x04, 0x03, 0x03, 0x02, 0x02, 0x01, 0x01, 0x01,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01,
    0x01, 0x01, 0x02, 0x02, 0x03, 0x03, 0x04, 0x04,
    0x05, 0x06, 0x06, 0x07, 0x08, 0x08, 0x09, 0x0A,
    0x0B, 0x0C, 0x0D, 0x0E, 0x0F, 0x10, 0x11, 0x12,
    0x13, 0x14, 0x15, 0x17, 0x18, 0x19, 0x1A, 0x1B,
    0x1D, 0x1E, 0x1F, 0x21, 0x22, 0x24, 0x25, 0x26,
    0x28, 0x29, 0x2B, 0x2C, 0x2E, 0x2F, 0x31, 0x32,
    0x34, 0x35, 0x37, 0x38, 0x3A, 0x3B, 0x3D, 0x3E,
};

/* RAM definition */
unsigned short inc1;                                    /* Incremental width (LOW) */
unsigned short inc2;                                    /* Incremental width (HIGH) */
unsigned short pos1;                                    /* Table position (LOW) */
unsigned short pos2;                                    /* Table position (HIGH) */

int key_data, old_key;                                  /* Input key code new & old */
int i,j;                                                /* Loop counter */
int key;                                                /* Input key code */

/* Vector address */
#pragma section V1                                      /* Set vector section */
void (*const VEC_TBL1[])(void) = {
    INIT                                                /* H'0000 Reset vector */
};

#pragma section V2                                      /* Set vector section */
void (*const VEC_TBL2[])(void) = {
    tmrw                                                /* H'002a Timer W interrupt vector */
};
#pragma section                                         /* P */
```

```
/*********************************************************/
/* Main program                                          */
/*********************************************************/
void main(void)
{
    set_imask_ccr(1);                           /* CCR I-bit = 1 */

    pos1 = 0;                                   /* Clear table position (LOW) to zero        */
    pos2 = 0;                                   /* Clear table position (HIGH) to zero       */
    inc1 = 0;                                   /* Clear incremental width (LOW) to zero     */
    inc2 = 0;                                   /* Clear incremental width (HIGH) to zero    */


    PCR1 = 0x00;                                /* Port 1 0-2 bits input */

        PCR7  = 0xff;                           /* Port 7 0-3 bits output */

    PWMB = 1;                                   /* PWM output */
    TCRW = 0x82;                                /* Initialize TCRW    */
    GRA = 0xff;                                 /* General-purpose register A */
    GRB = 0x00;                                 /* General-purpose register B */
    TCNT = 0;                                   /* Initialize timer count register    */
    IMIEA = 1;                            /* Input-capture/compare-match interrupt enable A */


    key_data = old_key = -1;                    /* Initialize key data */
    while(1){
        key_data = input_key();                 /* Input key */
        if ( key_data != -1) {                  /* KEY ON      */
            if ( old_key != key_data ) {
                inc1 = PB[key_data][0];         /* Set increase data (LOW) */
                inc2 = PB[key_data][1];         /* Set increase data (HIGH) */
                TCNT = 0;                       /* Initialize timer count register    */
                CTS = 1;                        /* Count register start       */
                set_imask_ccr(0);               /* CCR I-bit = 0 */
                old_key = key_data;             /* Save current key data */
                for(i = 0;i < 100;i++);         /* Dummy loop  */
            }
        } else {                                /* KEY OFF     */
            set_imask_ccr(1);                   /* CCR I-bit = 1 */
            CTS = 0;                            /* Count register stop */
            pos1 = 0;                           /* Clear table position (LOW) to zero        */
            pos2 = 0;                           /* Clear table position (HIGH) to zero       */
            inc1 = 0;                           /* Clear incremental width (LOW) to zero */
            inc2 = 0;                           /* Clear incremental width (HIGH) to zero */
            old_key = -1;                       /* Clear old input data */
        }
    }
}
```

```
/*********************************************************/
/* Input key data                                        */
/*********************************************************/
int input_key(void)
{
    key = 0;                                          /* Set start key code   */
    for ( j=0x01; j<=0x08; j<<=1 ) {
        PDR7 = j;                                     /* Output level to line */
        for ( i=0x01; i<=0x04; i<<=1 ) {
            if ( PDR1 & i ) {                         /* Read level from line */
                return key;
            }
            key++;
        }
    }
    return -1;
}

/*********************************************************/
/* Timer W interrupt                                     */
/*********************************************************/
void tmrw(void)
{
   if(IMFA == 1){                                     /* IMFA = 1 ? */
        IMFA  = 0;                                    /* Clear IMFA to 0 */
   }
    if(IMFB == 1){                                    /* IMFB = 1 ? */
        IMFB  = 0;                                    /* Clear IMFB to 0 */
   }

    pos1 += inc1;                                     /* Next position (LOW) */
    pos2 += inc2;                                     /* Next position (HIGH) */

               /* Sine table lookup  */
    GRB = sine[pos1 >> 8] + sine[pos2 >> 8];
}
```

Website and Support

Renesas Technology Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry
csc@renesas.com

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Dec.20.03	—	First edition issued
2.00	Sep.01.06	1, 9	New addition and content correction

Keep safety first in your circuit designs!

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

Notes regarding these materials

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