

R8C/11 GROUP

SAMPLE PROGRAM MUSICAL SCALE PROGRAM

REJ05B0273-0110Z

Rev.1.10

Nov 10, 2003

1. Abstract

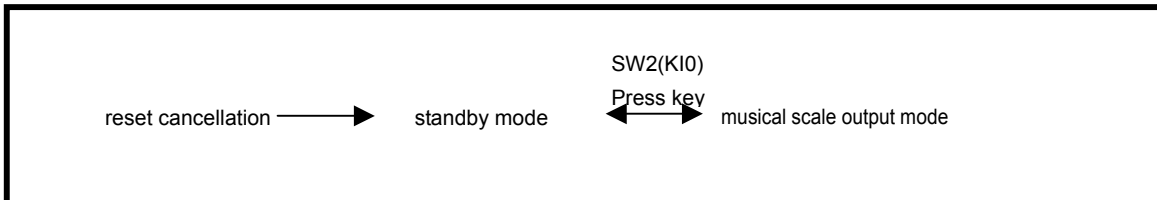
The musical scale program uses blow functions.

- Timer X(timer mode)
- Timer Z(programmable waveform generation mode)
- A-D converter (repeat mode)

2. Introduction

Operation (1) The musical scale program is divided by two mode

- Standby mode
- Musical scale output mode



(2) After the reset cancellation, the microcomputer changes to standby mode.

(3) When pressing SW2(KIO) in standby mode, the microcomputer changes to musical scale output mode.

(4) When pressing SW2(KIO) in musical scale output mode, the microcomputer changes to standby mode.

(5) The buzzer outputs bass C→D→E→F→G→A→B→octave C→silence(500 ms)→octave C→B→A→G→F→E→D→bass C→silence(500 ms)→bass C→D→E repeatedly in musical scale output mode.

(6) The musical scale output changes every one second. The musical scale frequency shows as below (Table1).

Table 1 :

Musical scale	Bass (C)	(D)	(E)	(F)	(G)	(A)	(B)	Octave (C)
Frequency(Hz)	523.25	587.32	659.25	698.45	783.98	880.00	987.76	1046.50

(7) The duty ratio of the buzzer output depends on the A-D value which is determined at the following (9).

Table 2 :

A-D determinate value	00~1F	20~3F	40~5F	60~F	80~9F	A0~BF	C0~DF	E0~FF
Duty ratio(%)	25.00	33.75	42.50	50.00	56.25	62.50	68.75	75.00

(8) The analog input converts to digital with volume(VR1). The A-D converter uses repeat mode as the A-D input and the pin uses P07/AN0.

(9) The A-D converter samples A-D convert value of P07/AN0 four times by 10 ms periods and uses its average value.

The calculating method shows as below.

1. The A-D converter adds the A-D value which is sampled on the A-D work space.
2. When the A-D converter samples four times, it sets lower 2 bits of the A-D work space value to "0"

3. Upper 8 bits left in the A-D work space value is applied to the average value.
 (10) The key input samples by 10 ms periods and confirms by three times match. (chattering removal)
 (11) The input key, SW2(KI0) is L active (L : with pressing H : without pressing)
 (12) In standby mode and musical scale output mode, the LED indicates as below. (Table 3)

Table 3 :

	LED1(Red)	LED2(Green)	LED3(Green)	LED4(Green)
In standby mode	Turn on	Turn off	Turn off	Turn off
In musical scale mode	Turn off	Turn on	Turn on	Turn on

(13) The buzzer output uses the timer Z programmable waveform mode and outputs from P31 P31(TZOUT).

(14) The register value of the timer Z which uses for the buzzer output is set as below.
 (The value in below table 4 is set as close to table 1 and table 2)

Table 4.1 :

Musical scale	Prescaler value	Setting time (us)	Timer sum (tzpr+tzsc)	Output time (ms) (tzpr+tzsc)	Output frequency (Hz)
Bass (C)	60	24	80	1.92	520.8333
(D)	53	21.2	80	1.696	589.6226
(E)	47	18.8	80	1.504	664.8936
(F)	45	18	80	1.44	694.4444
(G)	40	16	80	1.28	781.2500
(A)	36	14.4	80	1.152	868.0556
(B)	32	12.8	80	1.024	976.5626
Octave (C)	30	12	80	0.96	1041.6667

Table 4.2 :

Timer Z value/ A-D value range	00~ 1F	20~ 3F	40~ 5F	60~ 7F	80~ 9F	A0~ BF	C0~ DF	E0~ FF
Primary (tzpr)	20	27	34	40	45	50	55	60
Secondary (tzsc)	60	53	46	40	35	30	25	20

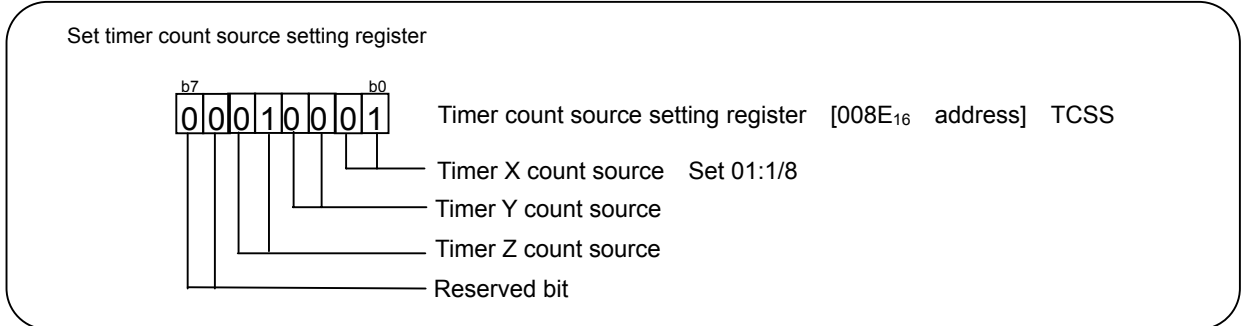
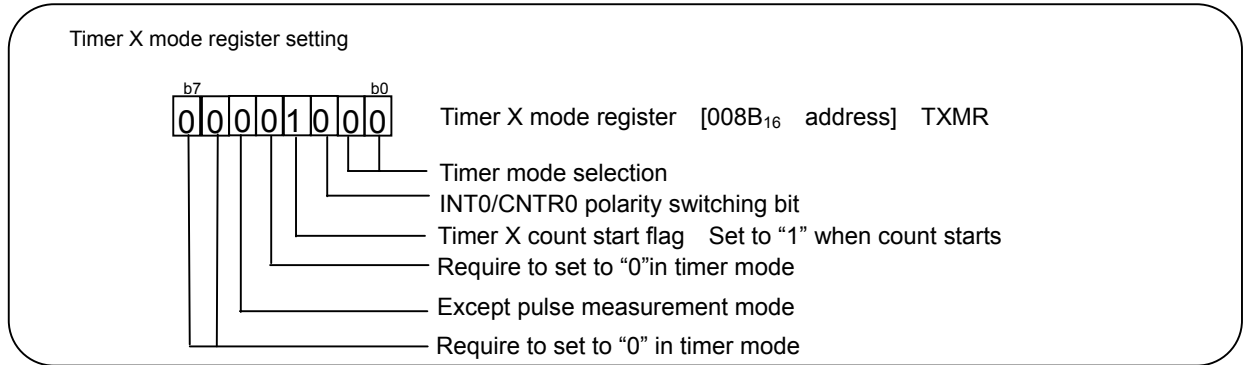
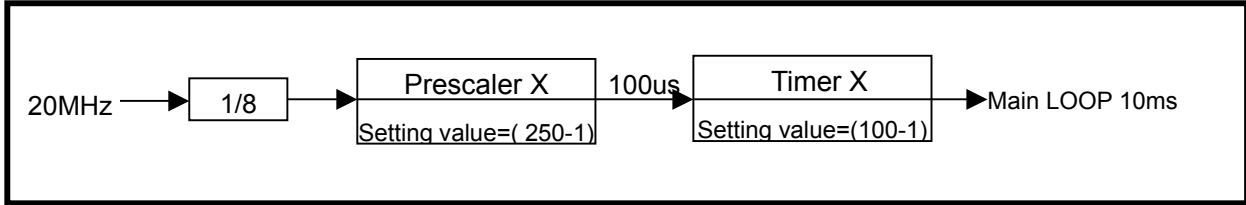
Musical scale/ A-D value range/ Output time ms		00~ 1F	20~ 3F	40~ 5F	60~ 7F	80~ 9F	A0~ BF	C0~ DF	E0~ FF
Bass (C) Prescaler 48	tzpr	0.480	0.648	0.816	0.960	1.080	1.200	1.320	1.440
	tzsc	1.440	1.272	1.104	0.960	0.840	0.720	0.600	0.480
(D) Prescaler 43	tzpr	0.424	0.572	0.721	0.848	0.954	1.060	1.166	1.272
	tzsc	1.272	1.124	0.975	0.848	0.742	0.636	0.530	0.424
(E) Prescaler 38	tzpr	0.376	0.508	0.639	0.752	0.846	0.940	1.034	1.128
	tzsc	1.128	0.996	0.865	0.752	0.658	0.564	0.470	0.376
(F) Prescaler 36	tzpr	0.360	0.486	0.612	0.720	0.810	0.900	0.990	1.080
	tzsc	1.080	0.954	0.828	0.720	0.630	0.540	0.450	0.360
(G) Prescaler 32	tzpr	0.320	0.432	0.544	0.640	0.720	0.800	0.880	0.960
	tzsc	0.960	0.848	0.736	0.640	0.560	0.480	0.400	0.320
(A) Prescaler 28	tzpr	0.288	0.389	0.490	0.576	0.648	0.720	0.792	0.864
	tzsc	0.864	0.763	0.662	0.576	0.504	0.432	0.360	0.288
(B) Prescaler 25	tzpr	0.256	0.346	0.435	0.512	0.576	0.640	0.704	0.768
	tzsc	0.768	0.678	0.589	0.512	0.448	0.384	0.320	0.256
Octave (C) Prescaler 24	tzpr	0.240	0.324	0.408	0.480	0.540	0.600	0.660	0.720
	tzsc	0.720	0.636	0.552	0.480	0.420	0.360	0.300	0.240

(15) When the A-D value range crosses over during the musical scale output, the register is reset by the new A-D value range.

3. Program reference

3.1 Timer X (Timer mode)

This sample program uses the timer X (timer mode) and stabilizes the main LOOP period.
The setting value of the timer X shows as below.

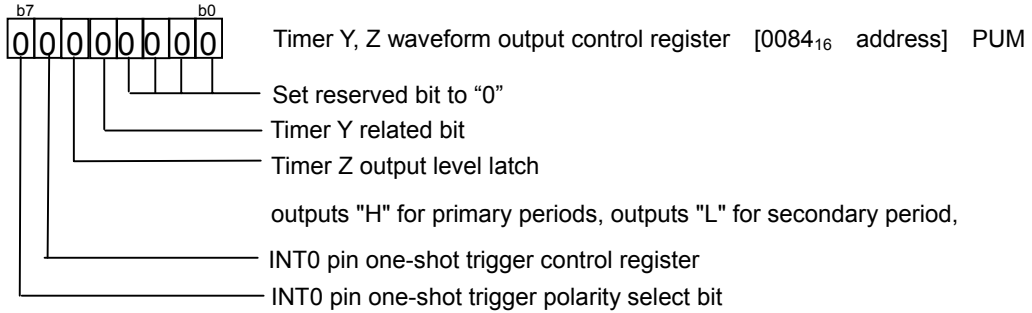


When setting as above, the timer X interrupt request bit is set to "1" by 10 ms periods.
Before this sample program executes process, it confirms to pass 10 ms by the timer X interrupt request bit.
When it confirms to pass 10 ms, it sets the timer X interrupt request bit to "0" and executes main process.
When it doesn't confirm to pass 10 ms, it waits for the timer X interrupt request bit to be set to "1".

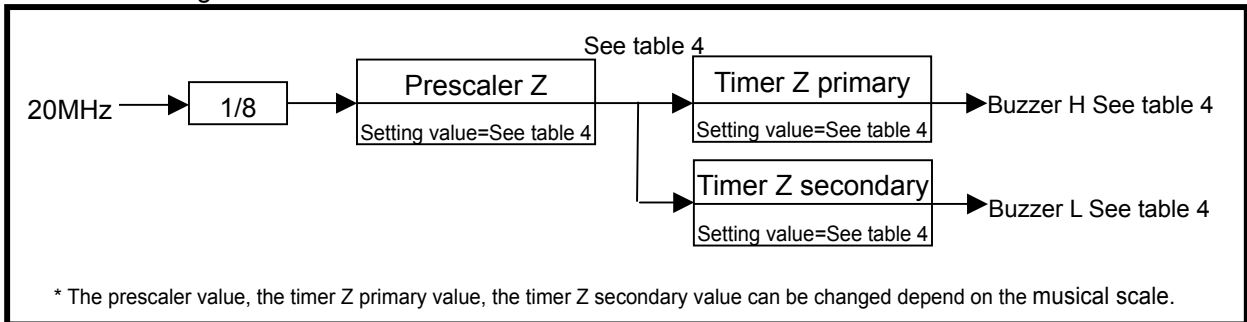
3.2 Timer Z (programmable waveform generation mode)

The buzzer makes sounds using the timer Z (programmable waveform generation mode). When setting as below, the musical scale is enabled to output. This program sets output level in the timer Y and the timer Z output level latch (TZOPL) of the Z waveform output control register (PUM). Setting the timer Z output level latch to "0" causes outputs "H" for primary period, outputs "L" for secondary period and outputs "L" when the timer is stopped.

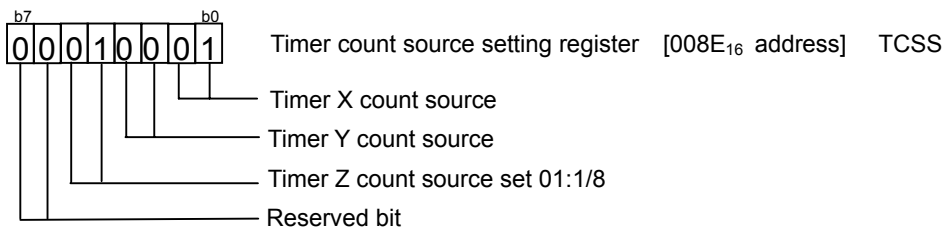
Setting of Timer Y, Z waveform output control register



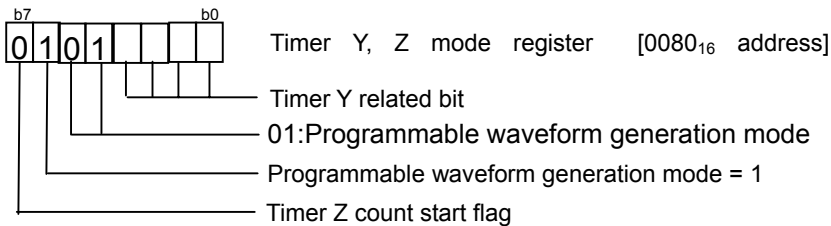
When this function starts counting, it outputs "H" for primary period and "L" for secondary period from P31 (TZOUT). When this function stops counting, it outputs "L" from P31 (TZOUT). The setting value of the timer Z shows as below.



Setting of timer count source setting register



Timer Y, Z mode register setting



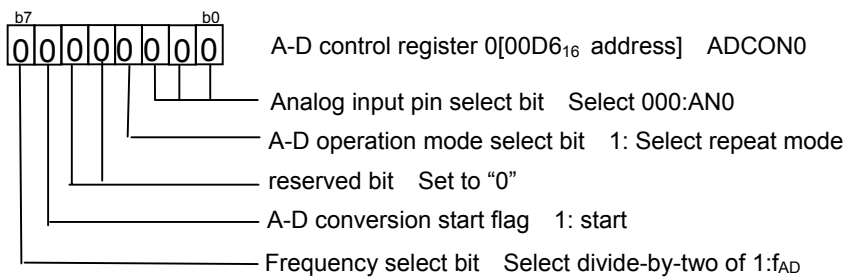
3.3 A-D conversion (repeat mode)

The musical scale program uses P07/AN0 in repeat mode..

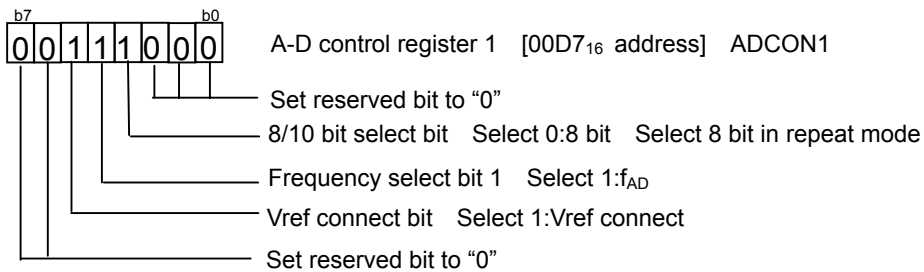
This sample program reads A-D register by main period (10ms) and confirms the A-D value four times on average. This sample program determines which area divided by eight the confirmed value belongs to and it sets the data value of its area to the timer Z (timer Z primary ·secondary) This determines the duty ratio.

When the SFR is reset, this program determines to convert which port inputs from, whether the input converts by 8/10 bit units and sets the use frequency.

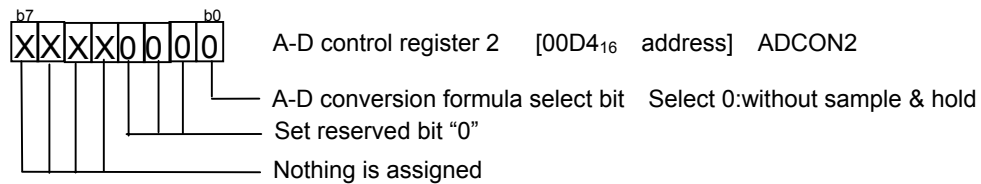
A-D control register 0 setting



A-D control register 1 setting



A-D control register 2 setting

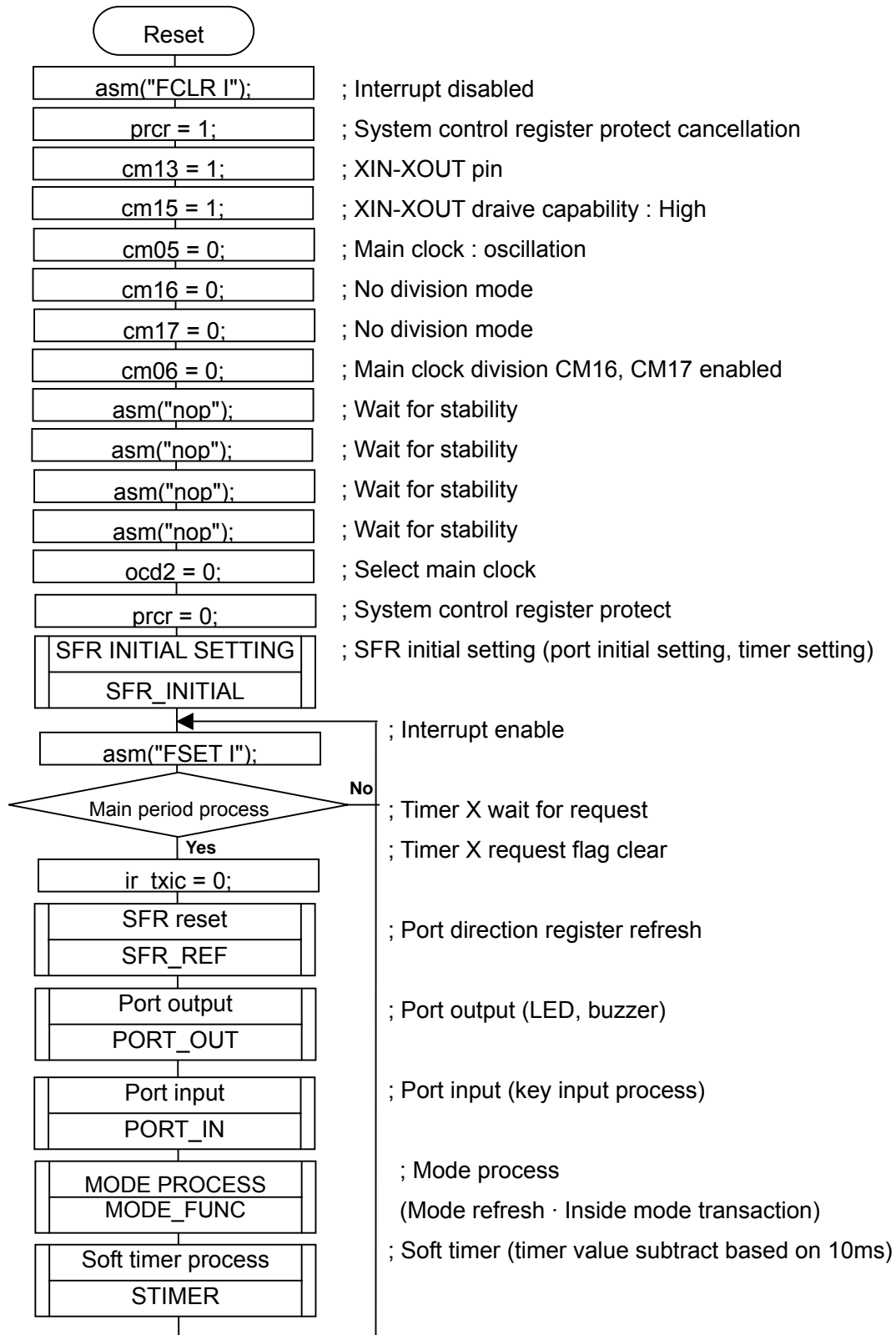


3.4 Chattering Removal

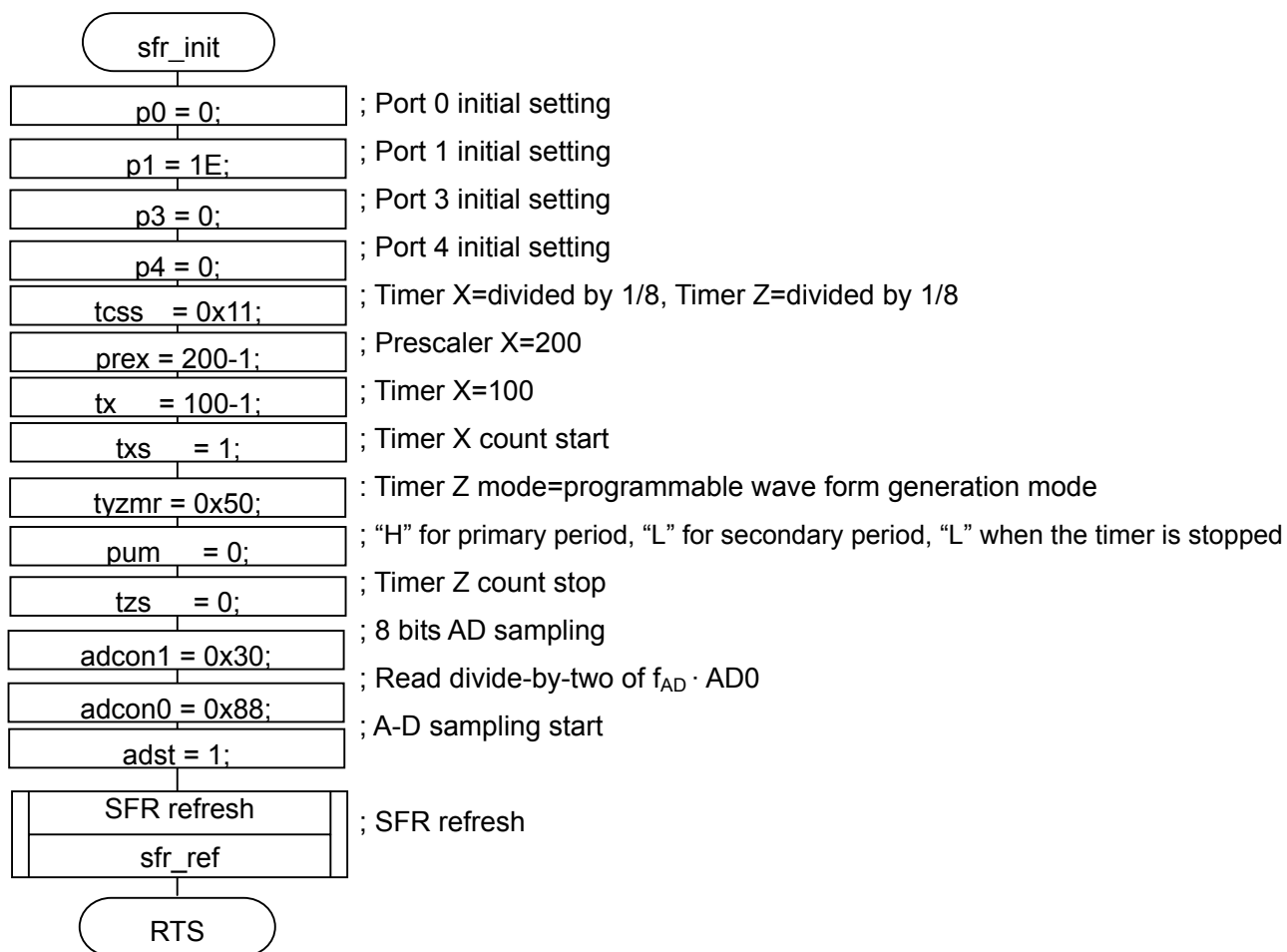
To confirm the level match needs to read the input port (SW2, SW3) a few time for the countmeasure against noise. The musical scale program samples by 10 ms periods and confirms the SW press three times matching.

4. Flow chart

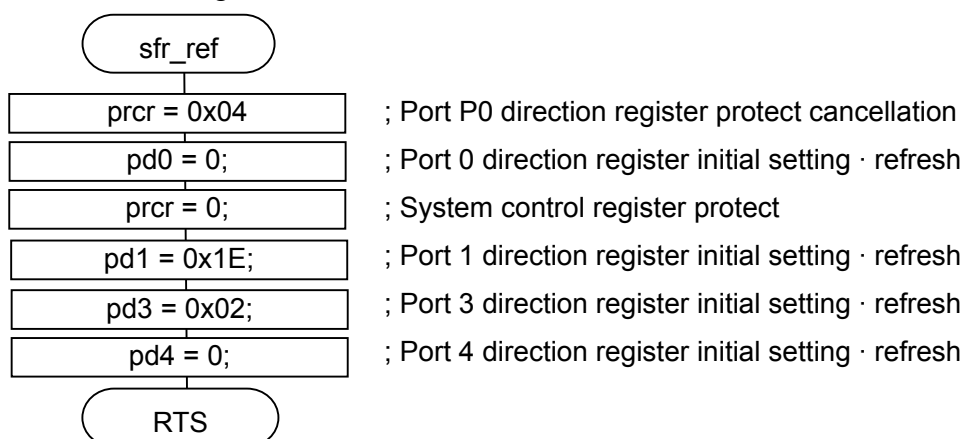
4.1 Initial operation and main LOOP



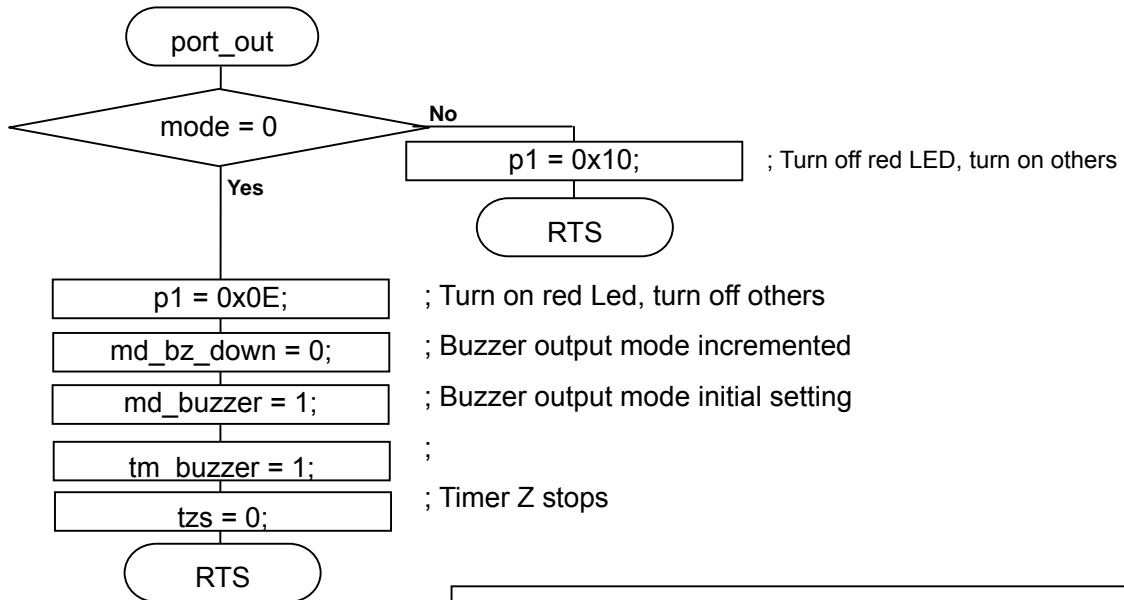
4.2 SFR initial setting



4.3 SFR initial setting



4.4 Port output



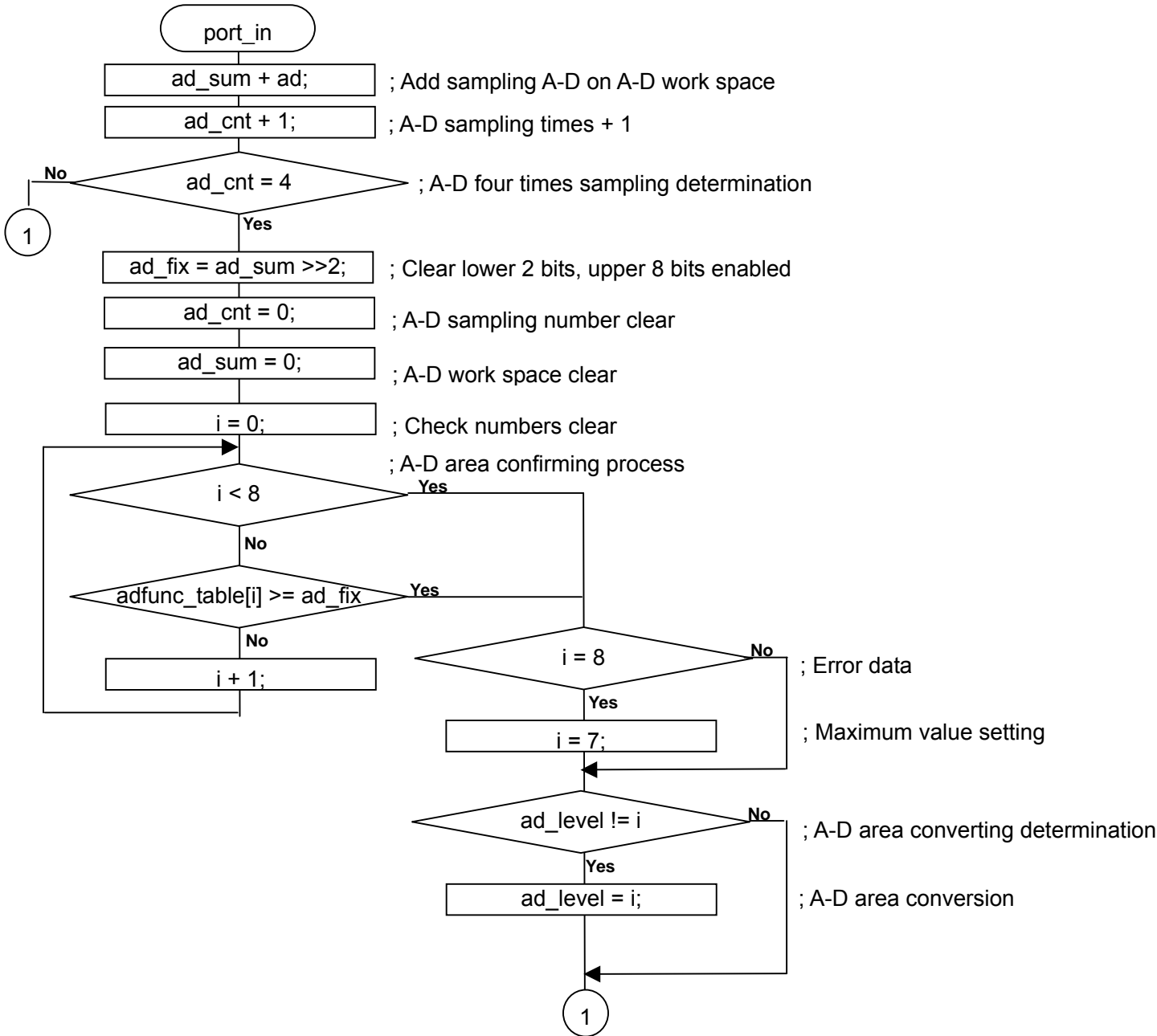
Timer Z prescaler value table :

bz_preztable[8] = {60,53,47,45,40,36,32,30};

Timer Z primary value table

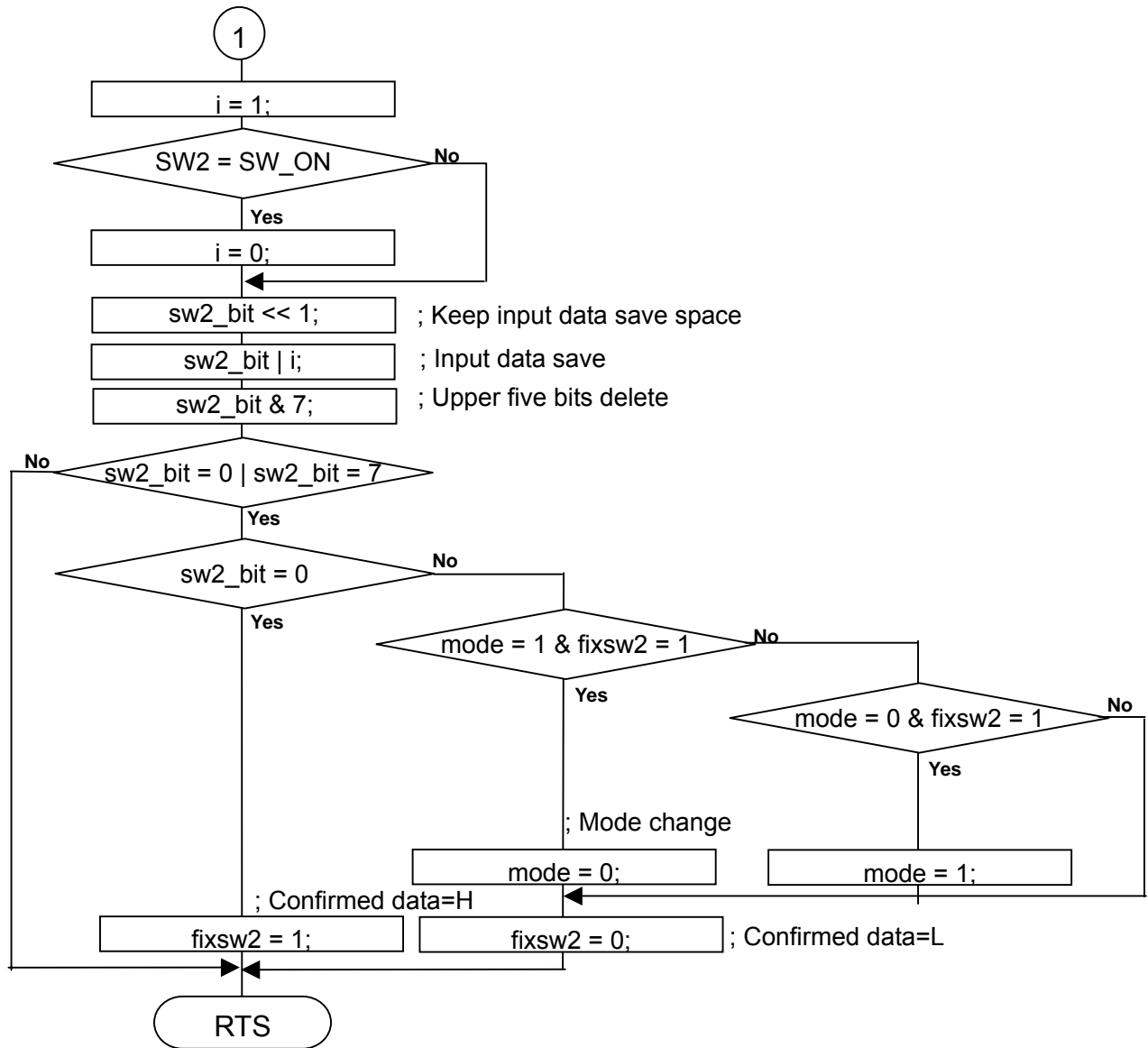
bz_tzsctable[8] = {20,27,34,40,45,50,55,60};

4.5 Port input

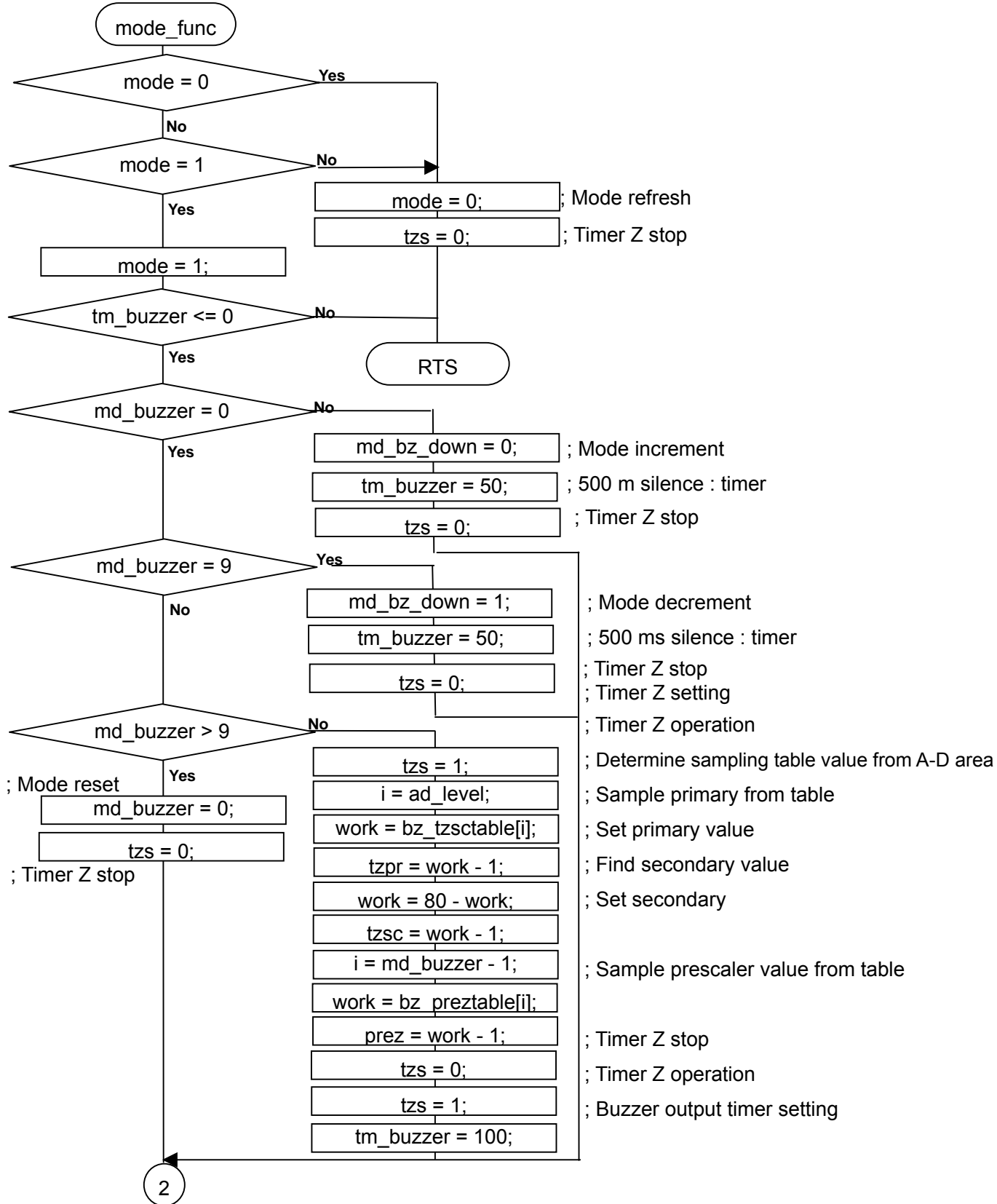


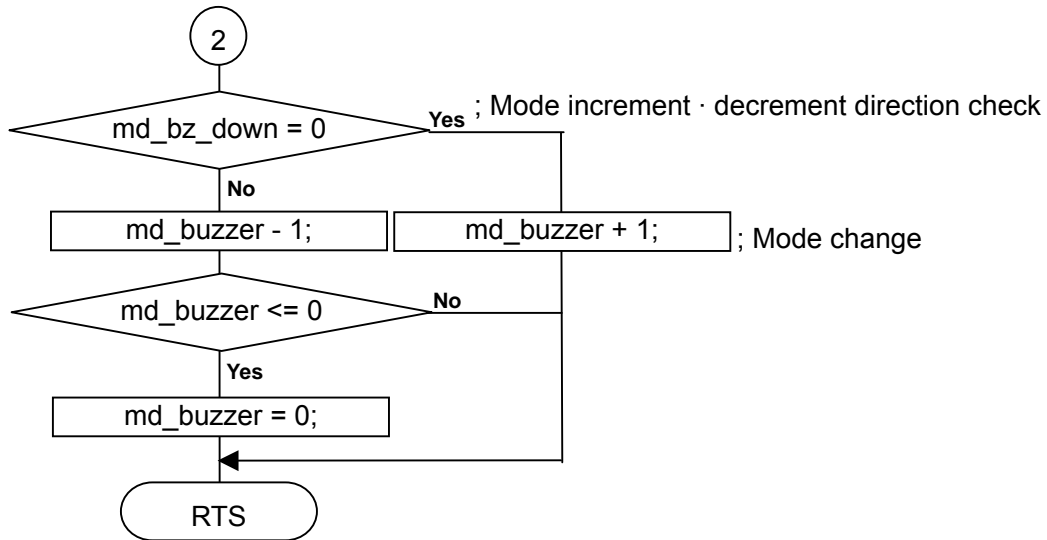
```

ad threshold table
adfunc_table[8]=
{0x1F,0x3F,0x5F,0x7F,0x9F,0xBF,0xDF,0xFF};
  
```

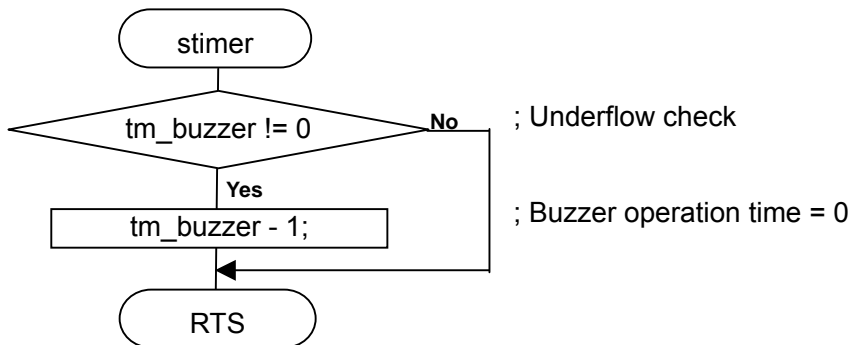


4.6 Mode process





4.7 Soft timer process



5. Program

```

/*****
*
*   File Name      : ad_onkai.h
*   Contents       : definition of R8C/11 Group SFR
*   Copyright,    : 2003 RENESAS TECHNOLOGY CORPORATION
*                  AND RENESAS SOLUTIONS CORPORATION
*   Version       : 1.00
*   note          :
*
*****/

/* Definition of RAM area */
char mode        = 0;          /* Mode number */
char sw2_bit     = 0;          /* Input SW2 data */
char fixsw2     = 1;          /* Input SW2 Settlement data */
char ad_cnt      = 0;          /* Number of A-D value sampling */
char md_bz_down = 0;          /* Buzzer output counting direction */

unsigned int md_buzzer = 0;    /* Buzzer output mode */
unsigned int tm_buzzer = 0;    /* Buzzer control timer */
unsigned int ad_fix    = 0;    /* A-D fixed data */
unsigned int ad_sum    = 0;    /* A-D calculation result data */
unsigned int ad_level  = 0;    /* Output range */

/* A-D table */
unsigned int adfunc_table[8] = {0x1F,0x3F,0x5F,0x7F,0x9F,0xBF,0xDF,0xFF};
/* Prescaler Z table */
unsigned int bz_preztable[8] = {60,53,47,45,40,36,32,30};
/* timer Z secondary register table */
unsigned int bz_tzsctable[8] = {20,27,34,40,45,50,55,60};

/* Declaration of function prototype */
extern void sfr_init(void);    /* Initial setting of SFR registers */
extern void sfr_ref(void);    /* Refresh of SFR registers */
extern void port_out(void);   /* Port output */
extern void port_in(void);    /* Port input */
extern void mode_func(void);  /* Mode processing */
extern void stimer(void);     /* Software timer */

/* Definition of base section */
#define SW_ON    0
#define SW_OFF  1
#define LOW     0
#define HIGH    1

/* Definition of port */
#define GREEN_LED2  p1_1
#define GREEN_LED1  p1_2
#define GREEN_LED0  p1_3
#define RED_LED     p1_4
#define SW2         p1_0
#define SW3         p4_5
#define BUZZER      p3_1

```

```

/*****
*
* File Name      : main.c
* Contents       : definition of R8C/11 Group SFR
* Copyright, 2003 RENESAS TECHNOLOGY CORPORATION
*                AND RENESAS SOLUTIONS CORPORATION
* Version        : 1.00
* note          :
*
*****/
#include "sfr_r811.h"      /* Definition of the R8C/11 SFR */
#include "ad_onkai.h"     /* Definition of processing for the musical scale
program */

main(){
    asm("FCLR I");        /* Interrupt disable */
    prcr = 1;             /* Protect off */
    cm13 = 1;             /* X-in X-out */
    cm15 = 1;             /* XCIN-XCOUT drive capacity select bit : HIGH */
    cm05 = 0;             /* X-in on */
    cm16 = 0;             /* Main clock = No division mode */
    cm17 = 0;
    cm06 = 0;             /* CM16 and CM17 enable */
    asm("nop");
    asm("nop");
    asm("nop");
    asm("nop");
    ocd2 = 0;             /* Main clock change */
    prcr = 0;             /* Protect on */

    sfr_init();          /* Initial setting of SFR registers */

    while(1){            /* Main processing */
        asm("FSET I");    /* Interrupt enable */
        while(ir_txic == 0){ /* Main cycle 10ms */
            ir_txic = 0;
            sfr_ref();     /* Refresh of SFR registers */
            port_out();    /* Port output */
            port_in();     /* Port input */
            mode_func();   /* Mode processing */
            stimer();      /* Software timer */
        }
    }
}

```

```

/*****
Name      : sfr_init
Parameters : None
Returns   : None
Description : Initial setting of SFR registers
*****/
void sfr_init(void){
    /* Setting port registers */
    p0 = 0;
    p1 = 0x1E;          /* p14-11 = H(Led 4.3.2.1) */
    p3 = 0;            /* p31 = L(Buzzer) */
    p4 = 0;

    tcss = 0x11;       /* division = X:1/8,Z:1/8 */
    /* Setting main cycle timer */
    /* 20MHz* 1/8 * 250 * 100 =10ms */
    prex = 250 - 1;    /* Setting Prescaler X register */
    tx = 100 - 1;      /* Setting timer X register */
    txs = 1;           /* Timer X count start flag = start */
    /* Setting Buzzer output timer */
    tyzmr = 0x50;      /* Timer Z mode = Programmable waveform generation mode */
    pum = 0;           /* Timer Y, Z waveform output control register */
    tzs = 0;           /* Timer Z count start flag = stop */

    /* A-D operation mode = Repeat mode */
    adcon1 = 0x30;     /* 8/10-bit mode select bit = 8-bit mode */
    adcon0 = 0x88;     /* fAD/2 and AN0 is selected */
    adst = 1;         /* A-D conversion started */

    sfr_ref();
}

/*****
Name      : sfr_ref
Parameters : None
Returns   : None
Description : Refresh of SFR registers
*****/
void sfr_ref(void){
    /* Setting port direction registers */
    prcr = 0x04;
    pd0 = 0;
    prcr = 0;
    pd1 = 0x1E;
    pd3 = 0x02;
    pd4 = 0;
}

```

```
/******  
Name      : port_out  
Parameters : None  
Returns   : None  
Description : Port output  
*****/  
void port_out(void){  
  
    if (mode == 0){  
        /* LED output processing */  
        pl = 0x0E;  
        /* Buzzer output processing */  
        md_bz_down = 0; /* Buzzer output counting direction = increment counting */  
        md_buzzer = 1; /* Buzzer output mode = C(1) */  
        tm_buzzer = 0; /* Buzzer control timer = 1 second */  
        tzs = 0;      /* Timer Z stops Counting */  
    }else{  
        /* LED output processing */  
        pl = 0x10;  
    }  
}
```

```

/*****
Name      : port_in
Parameters : None
Returns   : None
Description : Port input
*****/
void port_in(void){

    unsigned int i;

    /* A-D input */
    ad_sum = ad_sum + ad;          /* A-D calculation result data + picking A-D */
    ad_cnt = ad_cnt + 1;
    if (ad_cnt == 4){             /* Number of sampling become 4 */
        ad_fix = ad_sum >> 2;    /* 4 times shift to the right */
        ad_cnt = 0;              /* picking A-D space clear */
        ad_sum = 0;
        /* A-D Range settlement processing */
        i = 0;
        while (i < 8){           /* Output range is settled */
            if (adfunc_table[i] >= ad_fix){break;}
            i = i + 1;
        }
        if (ad_level != i){
            ad_level = i;        /* Change Output range */
        }
    }

    /* Determination of input level SW2 */
    i = 1;
    if (SW2 == SW_ON)i = 0;      /* Now determination SW2 */
    sw2_bit = sw2_bit << 1;     /* Check pulses matching a trigger input level 3
times */
    sw2_bit = sw2_bit | i;
    sw2_bit = sw2_bit & 7;
    if (sw2_bit == 0 || sw2_bit == 7){ /* Determinate input SW2 */
        if (sw2_bit == 0){
            /* Mode change */
            if (mode == 1 & fixsw2 == 1){
                mode = 0;        /* Setting the standby mode */
            }else{
                if (mode == 0 & fixsw2 == 1){
                    mode = 1;    /* Setting the buzzer output mode */
                }
            }
            fixsw2 = 0;          /* Input on */
        }else{
            fixsw2 = 1;         /* Input off */
        }
    }
}

```

```

/*****
Name      : mode_func
Parameters : None
Returns   : None
Description : Mode processing
*****/
void mode_func(void){

    unsigned int i,work;

    switch(mode){
    case 1:
        mode = 1; /* Set mode of the buzzer output mode */
        if (tm_buzzer <= 0){
            /* Buzzer output mode change */
            if (md_buzzer == 0){
                tm_buzzer = 50;          /* Buzzer control timer = 0.5 second */
                md_bz_down = 0;         /* Buzzer output counting direction =
increment counting */
                tzs = 0;                /* Timer Z stops Counting */
            }else if (md_buzzer == 9){
                tm_buzzer = 50;          /* buzzer control timer = 0.5 second */
                md_bz_down = 1;         /* Buzzer output counting direction =
decrement counting */
                tzs = 0;                /* Timer Z stops Counting */
            }else if (md_buzzer > 9){
                md_buzzer = 0;          /* Buzzer output mode = clear */
                tzs = 0;                /* Timer Z stops Counting */
            }else{
                /* output the musical scale now */
                tzs = 1;                /* Timer Z starts Counting */
                i = ad_level;
                work = bz_tzsctable[i];
                tzpr = work - 1;        /* setting of timer Z Primary register
*/

                work = 80 - work;
                tzsc = work - 1;        /* setting of timer Z secondary
register */

                i = md_buzzer - 1;
                work = bz_preztable[i]; /* setting of Prescaler Z register */
                prez = work - 1;
                tzs = 0;                /* Timer Z stops Counting */
                tzs = 1;                /* Timer Z starts Counting */
                tm_buzzer = 100;        /* buzzer control timer = 1 second */
            }
            if (md_bz_down == 0){        /* increment counting action */
                md_buzzer = md_buzzer + 1; /* increment counting of the timer */
            }else{                       /* decrement counting action */
                md_buzzer = md_buzzer - 1; /* decrement counting the timer */
                if (md_buzzer == 0){md_buzzer = 0;}
            }
        }
        break;
    default:
        mode = 0;                        /* Set mode of the standby mode */
        tzs = 0;                          /* Timer Z count start flag = stop */
        break;
    }
}

```

```
/* *****  
Name      : stimer  
Parameters : None  
Returns   : None  
Description : Software timer  
***** */  
void stimer(void){  
    /* Countdown of buzzer control timer */  
    if (tm_buzzer != 0){tm_buzzer = tm_buzzer - 1;}  
}
```

6. Reference Documents

Datasheet

R8C/11 Group Datasheet

(For the most current version, please visit Renesas Technology Home Page)

Hardware Manual

R8C/11 Group Hardware Manual

(For the most current version, please visit Renesas Technology Home Page)

7. Home Page and Support Information Window

Renesas Technology Home Page

<http://www.renesas.com/>

M16C Family MCU Technical Support Information Window

support_apl@renesas.com

DEVISION HISTORY	R8C/11 Group Musical Scale Program Application Note
------------------	--

Rev.	Date	Description	
		Page	Point
1.00	Sep 09, 2003	-	
1.10	Nov 10, 2003	ALL	For R8C/11 Group

Keep safety first in your circuit designs!

- 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

- 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.
- 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.
- 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 arising 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>).
- 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.
- 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.
- The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
- 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.
- Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.