お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

H8/300L SLP シリーズ

PWM モジュールによる DTMF 信号の生成 (DTMF)

要旨

DTMF (Dual Tone Multi-Frequency) は一般的に電話分野に応用されます。MCU に DTMF ジェネレータが内蔵されていないとき，MCU の PWM モジュールを使用してソフトウェアにより実現できます。

このアプリケーションノートでは 16 の異なるトーンを生成する方法を紹介します。各トーンは，一方は高周波数群から，もう一方は低周波数群から成る 2 つの周波数の合成です。

この技術は，他の合成信号を生成する場合にも使用できます。

動作確認デバイス

H8/38024

目次

1. 概要

DTMF には 16 の異なるトーンがあります。各音は，一方は高周波数群から，もう一方は低周波数群から成る 2 つの周波数の合成です。各群には 4 つの異なる周波数があります。

表 1　DTMF トーン群

群	周波数 (Hz)
低周波数群	● 697Hz ● 770Hz ● 852Hz ● 941Hz
高周波数群	● 1209Hz ● 1335Hz ● 1477Hz ● 1633Hz

DTMF トーンダイアルの周波数とキー配列は国際標準ですが，各周波数の許容範囲は国によって異なります。米国の規格では，ジェネレータで 1.5%，レシーバで 2%です。DTMF トーンは PWM モジュールとタイマ割り込みにより生成されるので，システムの水晶発振子の周波数の精度は，生成する周波数の精度に直接，影響します。システムで異なるクロック速度を使用するとき，ソースコードを調整する必要があります。

さらにトーンは，有効なダイアルデジットとして受け付けられるまで，ある一定の期間保持されなければなりません。米国では，デジットの最低保持期間は約 50ms であり，デジット間の間隔も約 50ms です。

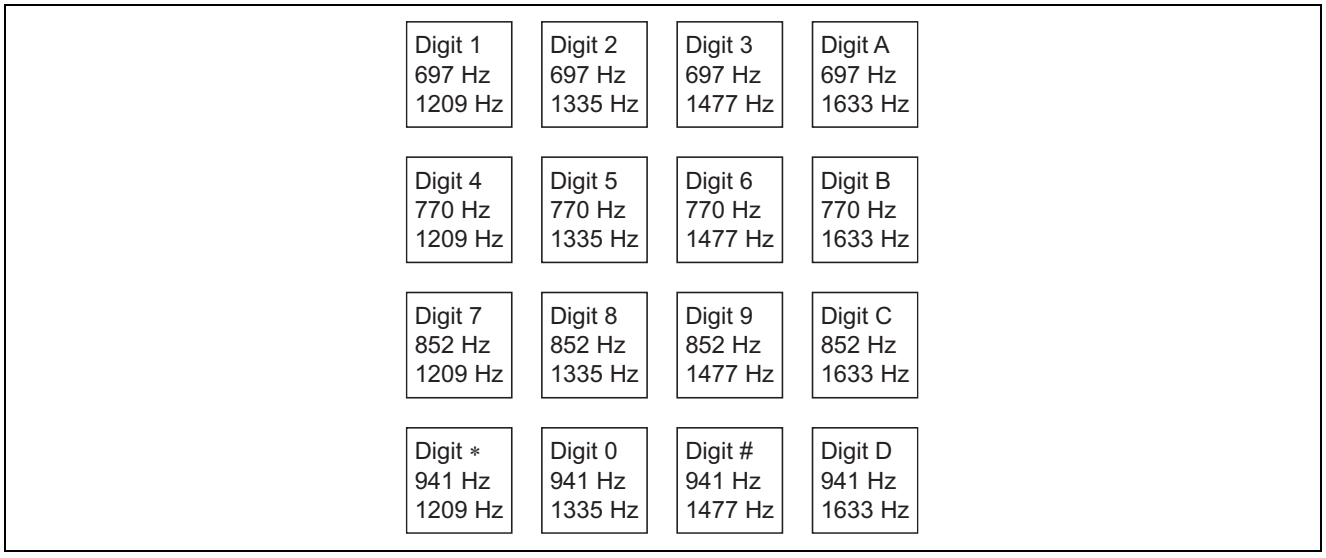

図1　キー配列とトーンを生成する 2 つの周波数

PWM モジュールから DTMF 信号を生成するには図 2 に示す 3 つの処理が必要です。

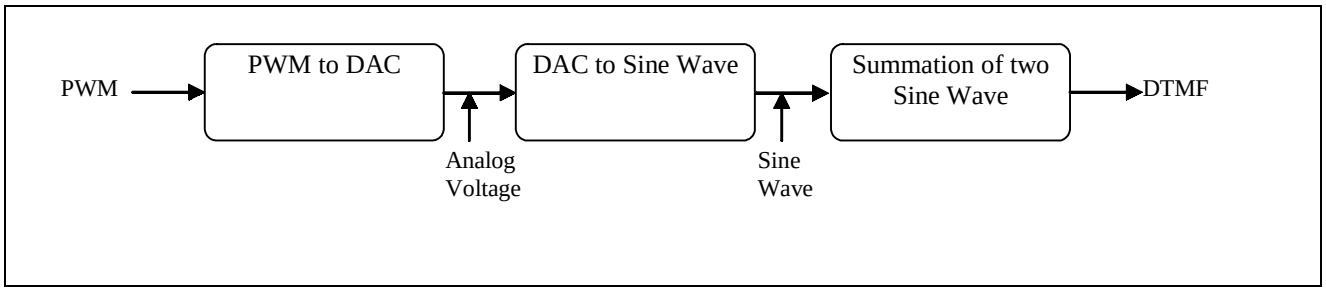

図2　DTMF 信号の生成処理の概要

1.1　PWM モジュールから D/A 変換器へ

PWM モジュールをローパスフィルタ (単純な RC 回路) に接続することにより，アナログ電圧を生成できます。その PWM 波形は，信号が High レベルのときにコンデンサは充電され，信号が Low レベルときにコンデンサは放電されます。図 3 は PWM モジュールから D/A 変換器までの処理動作です。D/A 変換器の説明の詳細は，アプリケーションノート AN: 03/03/004 – “PWM as a DAC”を参照してください。

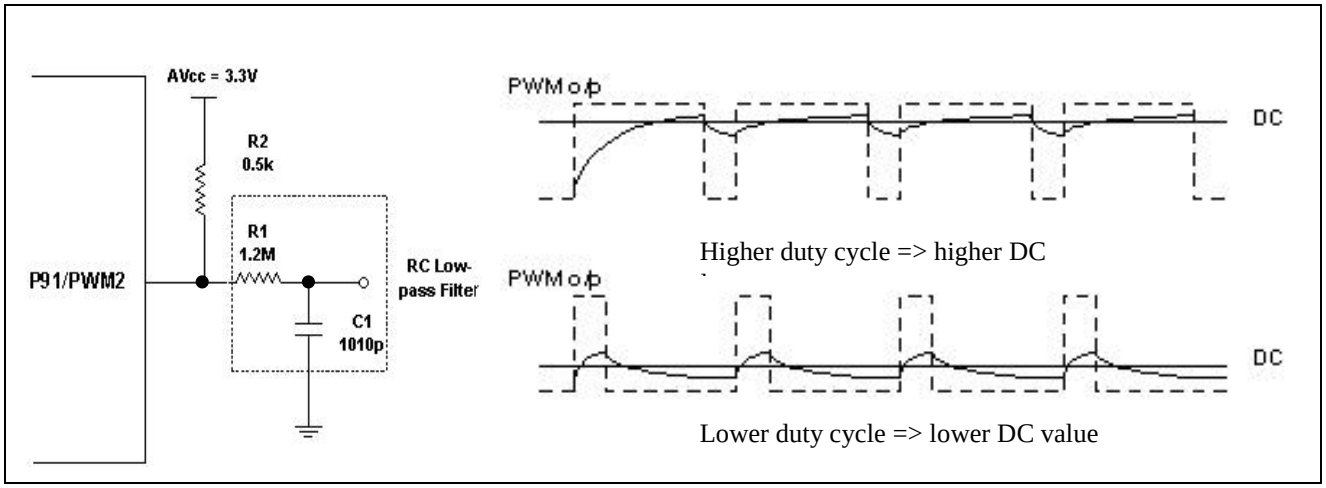

図3　PWM モジュールから D/A 変換器までの動作例

1.2 D/A 変換器からの正弦波

生成された DC 電圧レベルが正弦波である場合，以下の 2 つのパラメータを計算することが必要です。

- 信号の期間
- サンプリング期間

注：アプリケーションノート“PWM Sine Wave Generation”**を参照してください。**

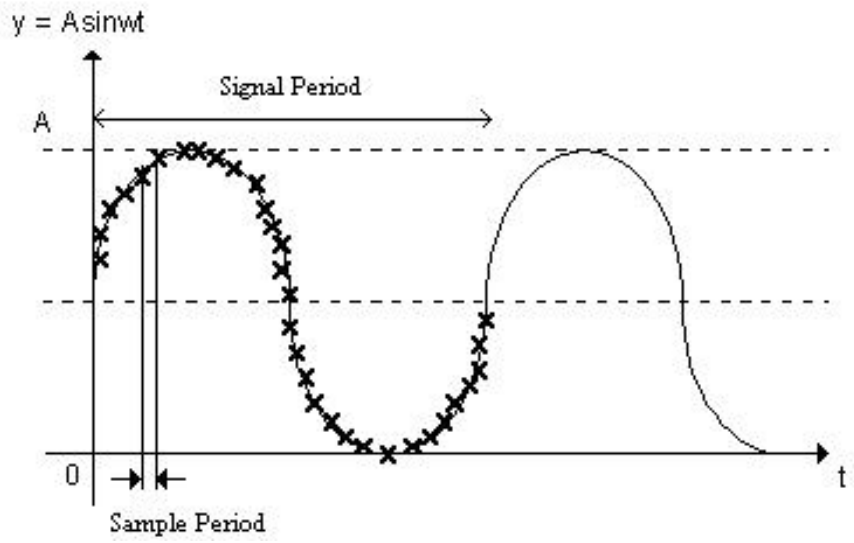

図4　正弦波の例

これは，信号の期間はユーザが望む周波数です。サンプリング期間を生成するために，非同期イベントカウンタ (Asynchronous Event Counter) が選択されます。AEC は 8 ビットのイベントカウンタで，入力クロックソースは∅/2 (∅ = 水晶発振子の周波数 / 2) に設定されます。AEC 割り込みが，入力クロックソースの 256 カウントごとに発生します。

たとえば，使用の水晶発振子が 9.8304 MHz のとき

AEC 割り込みが 1 回発生する周期を $T_{interrupt}$ とすると，

$T_{interrupt}$ = ((1 / (∅/2)) x 256 **回**
= (1 / (crystal / 4)) x 256 **回**
= (1 / (9.8304MHz / 4) x 256 **回**
= 10.42µs

サンプリング期間は，1 回の AEC 割り込みの発生に相当します。AEC 割り込みが発生するたびに割り込みサービスルーチン (Interrupt Service Routine) が計算したパルス幅を PWM 幅レジスタに格納します。

サンプリング周波数 = 1 / $T_{interrupt}$
= 9600Hz

パルス幅を計算するには，インクリメントカウンタの値が必要です。インクリメントカウンタの値は以下のように計算します。

仮定条件：

- 完全な正弦波の表のためのサンプリング数：256
- サンプリング周波数 = 9600Hz
- 信号周波数 = 852Hz

インクリメントカウンタの値 = 256/**インクリメント数**

インクリメント数はサンプリング周波数と信号周波数に依存します。また，インクリメント数は正弦波表内の 1 周期分のデータ数に等しくなります。

インクリメント数 = **サンプリング周波数/信号周波数**

∴ **インクリメントカウンタの値** = 256/(**サンプリング周波数/信号周波数**)

= 256 * **信号周波数/サンプリング周波数**

= 256 * (852Hz)/(9600Hz)

= 22.72

以上の計算はコンパイラによって演算されるので，他のパラメータを使用するときにはデフォルト値を変更してください。

表 2　インクリメントカウンタの値の計算

周波数 (Hz)	**インクリメントカウンタの値**
● 697Hz	● 18.587
● 770Hz	● 20.533
● 852Hz	● 22.720
● 941Hz	● 25.093
● 1209Hz	● 32.240
● 1335Hz	● 35.600
● 1477Hz	● 39.387
● 1633Hz	● 43.547

1.3　2 つの正弦波の合成

DTMF トーンダイアルには 2 種類の正弦波の生成が必要です。変調の基本的な概念は，2 つのカウンタを使用して異なる速度で正弦波表内のデータをインクリメントすることです。低周波数を生成するには，より小さなステップでインクリメントします。同様に高周波数を生成するには，より大きなステップでインクリメントします。その 2 つの正弦波は合成され，目的の DTMF 波形が生成されます。この処理を図 5 に示します。

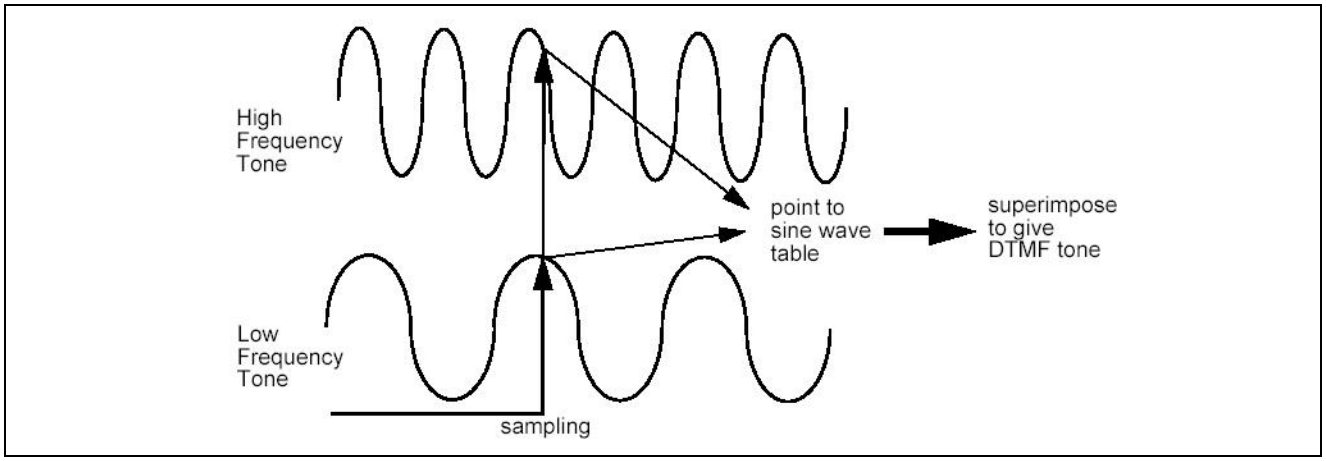

図5　2 つの正弦波の重畳

2. ハードウェアによる実現方法

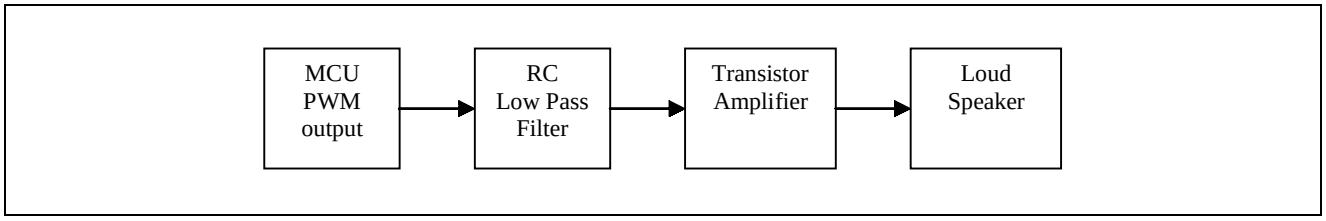

図6　DTMF 生成のブロック図

DTMF トーンは，SLP シリーズマイコンの PWM (Pulse Width Modulation) モジュールにおいて生成されます。ソフトウェアにより，正弦波の信号が，固定した周期のパルス幅が変化する連続したパルスに変調されます。パルスの変動量は，正弦波の電圧レベルのそれに対応します。PWM 出力端子にローパスフィルタ (Low Pass Filter) を外付けすることにより，PWM 信号が復調されます。LPF は連続したパルスをアナログのサイン信号に変換します。次に DTMF 波形はオーディオアンプに送られ，音が出力されます。

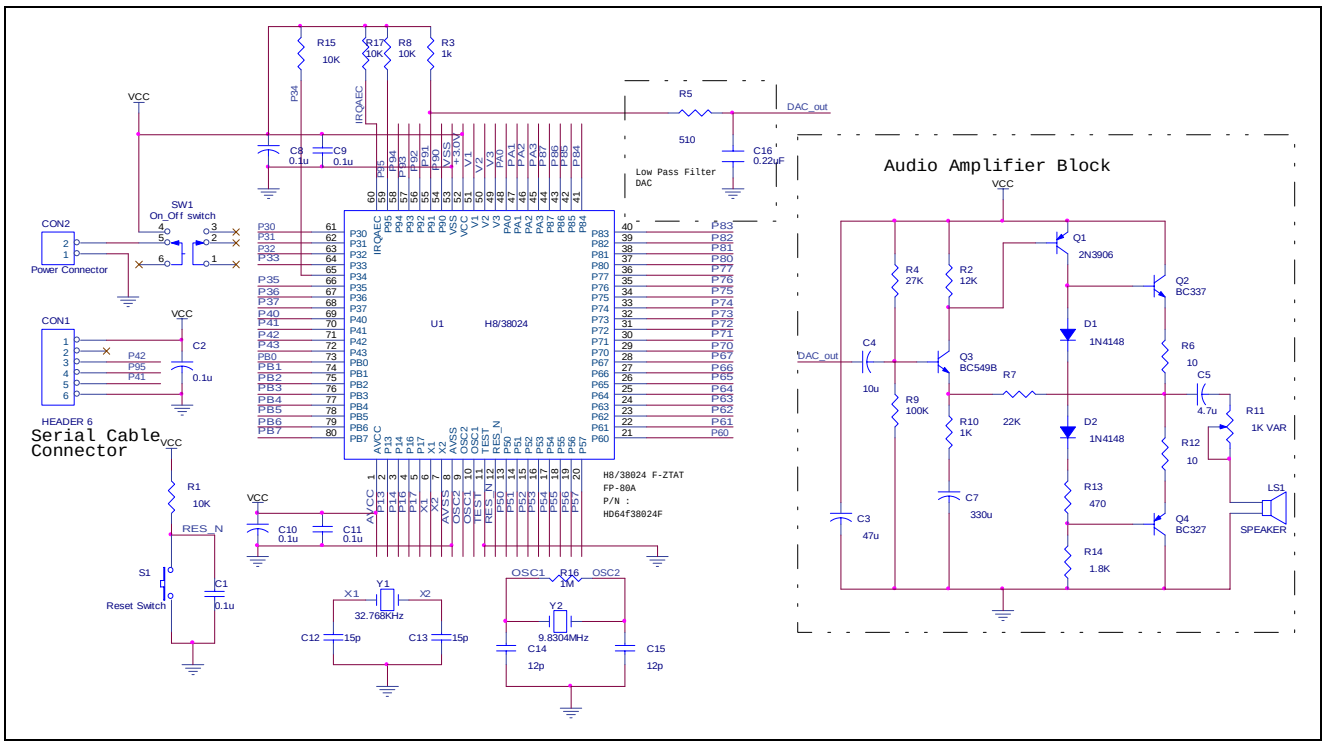

図7　DTMF 生成の概略

オーディオ信号の生成において，波形はグランドレベル，つまり 0V の平均値を持ちます。電圧は，正と負の値の間を変化します。しかし，PWM を使用したこのアプリケーションでは，電圧レベルは正の値しか持ちません。つまり，波形は DC のオフセット付きで波形が生成されます。ウォーミングアップ機能がこの問題を解決します。ソフトウェアのプログラムコードでオーディオ出力をウォーミングアップして，基準となるパルス幅がゆっくり 0 から 512 (10 ビット PWM の中間値) に増加するようにします。ウォーミングアップ時間が長いほど，波形は平滑になります。また，その逆も言えます。ユーザは，2 つのデジット間で PWM モジュールをイネーブルまたはディセーブルする代わりに，PWM 値を 512 に設定して D/A 変換器の出力電圧を 1.65V に保持することができます。

3. 動作と考察

ハードウェア回路によりフラッシュメモリへの書き込みができます。ユーザは PC のシリアルポートを介して DTMF ダイアルデモプログラムをダウンロードできます。ユーザプログラムをダウンロードするための PC アプリケーションソフト Flash Development Toolkit (FDT) は，以下の URL から無償でダウンロードできます。

www.eu.renesas.com .

DTMF ダイアルプログラムをダウンロードした後，MCU をリセットしてプログラムを実行してください。実行中，ユーザはスピーカから DTMF トーンを聞くことができるはずです。デモプログラムはすべてのデジットをダイアルした後に終了します。ユーザは MCU をリセットすることで，この処理を繰り返すことができます。

DTMF ダイアルデモプログラムは，内蔵の PWM モジュールを使い電話番号をダイアルします。ダイアルする電話番号は可変配列 PHONE_NO に格納されており，この例では「123456789」に設定されています。任意の長さの電話番号をダイアルすることができます。この例では，0 デジットから 9 デジットは 16 進数の H’00，H’01，…H’08，H’09 で表され，H’0A から H’0F はそれぞれ A， B，C，D， *，#に対応します。電話番号は H’10 バイトで停止します。

DTMF ダイアルデモプログラムは，#define 文の XTAL 値を変更すれば，他の水晶発振子でも使用できます。

例：

水晶発振子が = 9.8304MHz のとき	➔	#define XTAL	9830400L	(デフォルト)
水晶発振子が = 4MHz のとき	➔	#define XTAL	4000000L	

H8/38024F MCU には PWM チャネルが 2 つあるため，ユーザはどちらの PWM モジュールを使用するか，ソースコードをコンパイルする前に定義しなければなりません。

例：

PWM1 を使用するとき	➔	#define PWM_use	1	(デフォルト)
PWM2 を使用するとき	➔	#define PWM_use	2	

図 8 と図 9 に，テクトロニクス社の TDS3054 デジタルオシロスコープを使い電話番号 8 の DTMF 波形と周波数のスペクトラムを表示しました。図 8 では，S/N 比は約 30.4 dB です。図 9 に，電話番号 8 (852Hz と 1335Hz) のピーク周波数を示します。

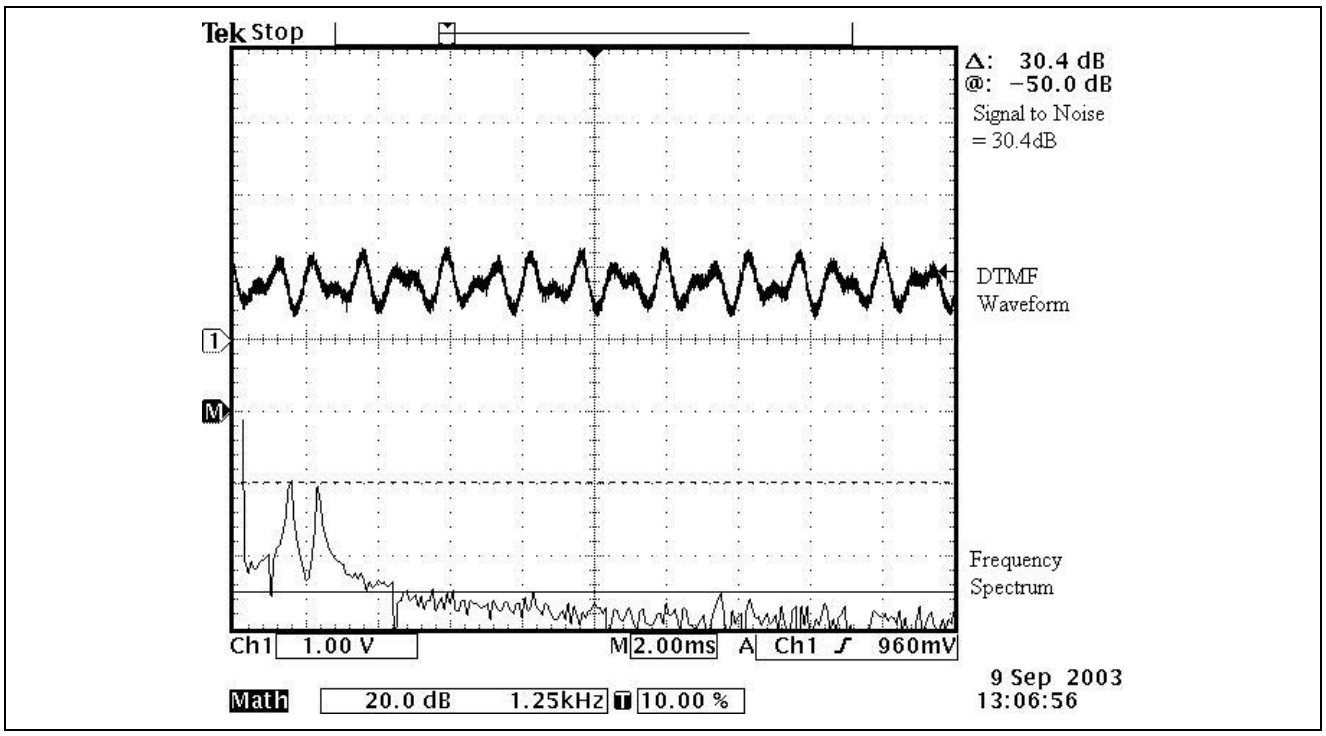

図8　8 デジットの DTMF 波形と周波数スペクトラム (SNR)

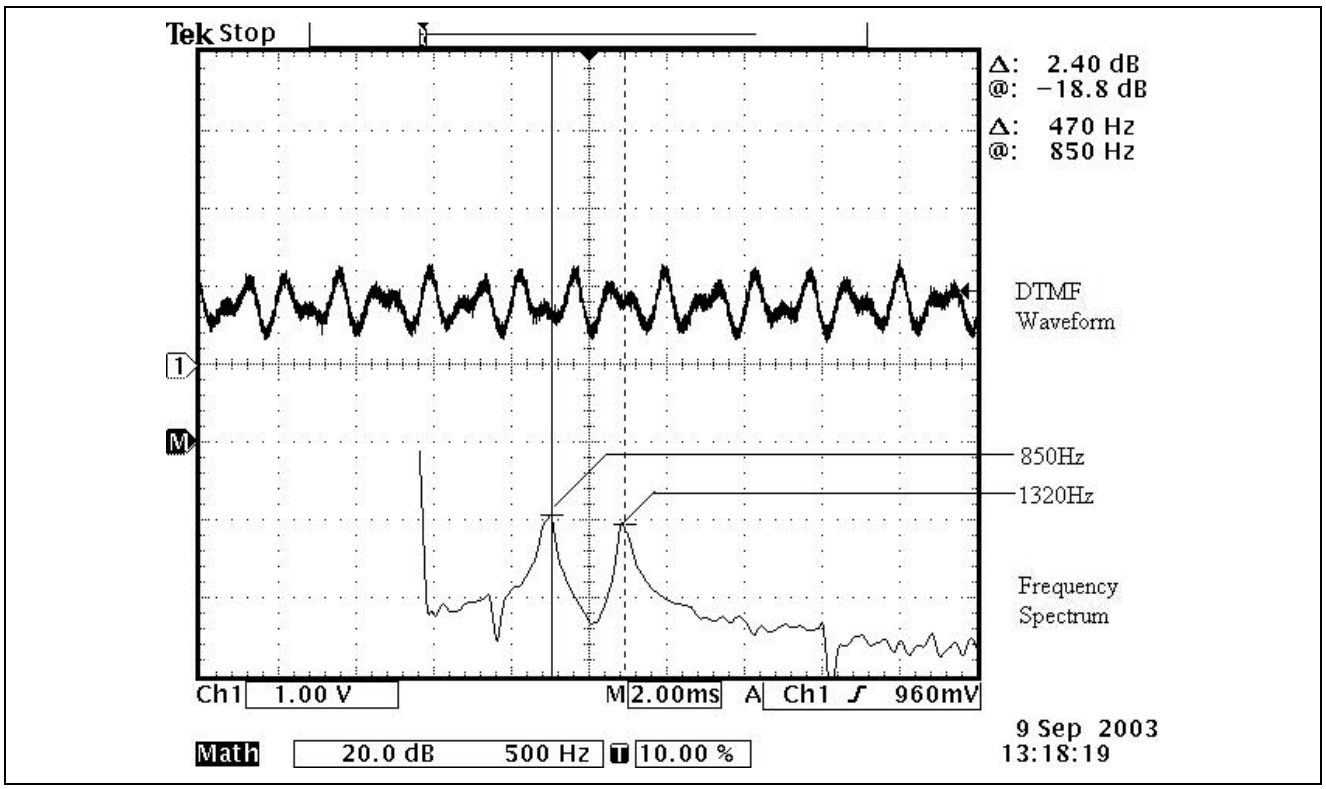

図9　デジット 8 に対する DTMF 波形と周波数スペクトラム (ピーク周波数)

4. プログラムリスト

以下が，H8/38024F SLP MCU 用の HEW プロジェクトジェネレータを使用して作成したプログラムコードです。ここでは無償の SLP/Tiny ツールチェーンを使用しています。

以下のプログラムリストは， “DTMF_dialing.c”の主要な機能です。

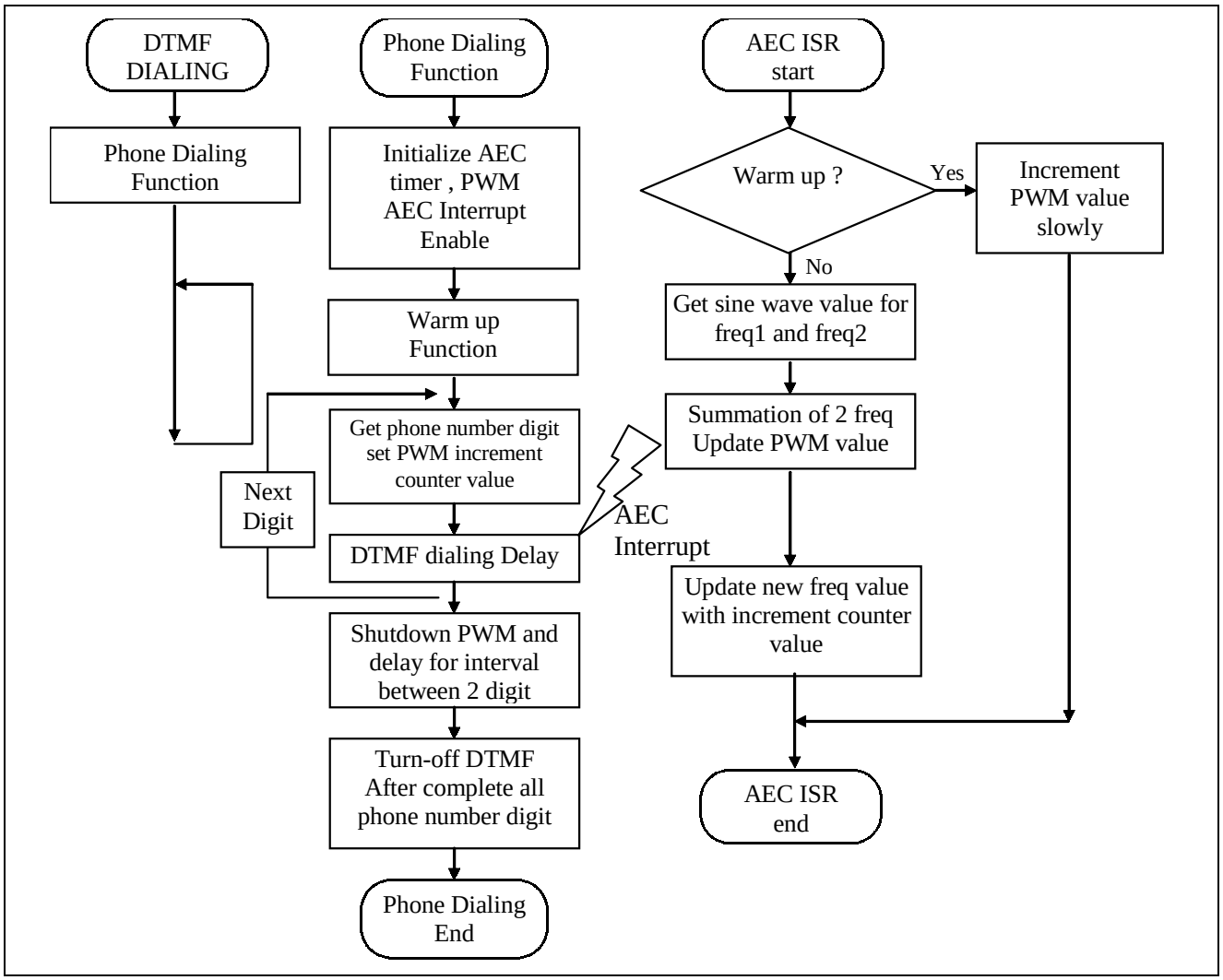

図10　DTMF_dialing.c のフローチャート

```
/*****************************************************************/
/*                                                               */
/*  FILE        :DTMF_Dialing.c                                  */
/*  DATE        :Tue, Mar 04, 2003                               */
/*  DESCRIPTION :Main Program                                    */
/*  CPU TYPE    :H8/38024F                                       */
/*                                                               */
/*  This file is generated by Renesas Project Generator (Ver.2.1).*/
/*                                                               */
/*****************************************************************/

/************************************************************/
/*  File Include                                            */
/************************************************************/
#include    <machine.h>
#include    "iodefine.h"
/************************************************************/
/* define                                                   */
/************************************************************/
#define  XTAL     9830400L
#define  sample_freq (XTAL/4L) / 256L //256 clock cycles per interrupt
#define  tone_low_a  ((256L * 697L)/100)/(sample_freq/100)
#define  tone_low_b  ((256L * 770L)/100)/(sample_freq/100)
#define  tone_low_c  ((256L * 852L)/100)/(sample_freq/100)
#define  tone_low_d  ((256L * 941L)/100)/(sample_freq/100)
#define  tone_high_a ((256L * 1209L)/100)/(sample_freq/100)
#define  tone_high_b ((256L * 1335L)/100)/(sample_freq/100)
#define  tone_high_c ((256L * 1477L)/100)/(sample_freq/100)
#define  tone_high_d ((256L * 1633L)/100)/(sample_freq/100)
#define  PWM_use     2       //select "1" for PWM channel 2
                             //select "0" for PWM channel 1
/************************************************************/
/*  Function define                                         */
/************************************************************/

void init_PWM(unsigned char);
void storeCount(unsigned short);
void aecint( void );
void init_AEC(void);
void init_DTMF(void);void off_DTMF(void);
void init_PWM1(unsigned char selClk1);
void init_PWM2(unsigned char selClk2);
void warm_up(void);
void dialing(void);

/************************************************************/
/*Constant Look up Table for Sine Wave value
/************************************************************/
const unsigned int   sample =sample_freq;
const unsigned int   Sine_Table[256]=
{
512,518,525,531,537,543,550,556,
562,568,574,580,586,592,598,604,
```

```
610,616,621,627,633,638,644,649,
654,659,664,669,674,679,684,688,
693,697,702,706,710,714,717,721,
725,728,731,734,737,740,743,746,
748,750,753,755,756,758,760,761,
762,763,764,765,766,766,766,767,
767,767,766,766,766,765,764,763,
762,760,759,757,755,754,751,749,
747,744,742,739,736,733,730,726,
723,719,715,712,708,704,699,695,
691,686,681,677,672,667,662,657,
652,646,641,635,630,624,619,613,
607,601,595,589,583,577,571,565,
559,553,546,540,534,528,521,515,
509,503,496,490,484,478,471,465,
459,453,447,441,435,429,423,417,
411,405,400,394,389,383,378,372,
367,362,357,352,347,343,338,333,
329,325,320,316,312,309,305,301,
298,294,291,288,285,282,280,277,
275,273,270,269,267,265,264,262,
261,260,259,258,258,257,257,257,
257,257,258,258,259,260,261,262,
263,264,266,268,269,271,274,276,
278,281,284,287,290,293,296,299,
303,307,310,314,318,322,327,331,
336,340,345,350,355,360,365,370,
375,380,386,391,397,403,408,414,
420,426,432,438,444,450,456,462,
468,474,481,487,493,499,506,512
};

const unsigned char tone[16][2] =
{                               //Keypad digit  Tone Pair (Hz)
{tone_low_d,tone_high_b},       //  '0'               941Hz, 1335Hz
{tone_low_a,tone_high_a},       //  '1'               697Hz, 1209Hz
{tone_low_a,tone_high_b},       //  '2'               697Hz, 1335Hz
{tone_low_a,tone_high_c},       //  '3'               697Hz, 1477Hz
{tone_low_b,tone_high_a},       //  '4'               770Hz, 1209Hz
{tone_low_b,tone_high_b},       //  '5'               770Hz, 1335Hz
{tone_low_b,tone_high_c},       //  '6'               770Hz, 1477Hz
{tone_low_c,tone_high_a},       //  '7'               852Hz, 1209Hz
{tone_low_c,tone_high_b},       //  '8'               852Hz, 1335Hz
{tone_low_c,tone_high_c},       //  '9'               852Hz, 1477Hz
{tone_low_a,tone_high_d},       //  'A'               697Hz, 1633Hz
{tone_low_b,tone_high_d},       //  'B'               770Hz, 1633Hz
{tone_low_c,tone_high_d},       //  'C'               852Hz, 1633Hz
{tone_low_d,tone_high_d},       //  'D'               941Hz, 1633Hz
{tone_low_d,tone_high_a},       //  '*'               941Hz, 1209Hz
{tone_low_d,tone_high_c},       //  '#'               941Hz, 1477Hz
};
```

```
/*********************************************************/
/*Global variable
/*********************************************************/
unsigned char PWDR_L2, PWDR_U2;
unsigned int i=0,j=0, count=0, inc1=0, inc2=0, final=0;
unsigned int lowcnt=0, hicnt=0;
unsigned char Ready = 0, DIGIT = 0;
unsigned char PHONE_NO[]={0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x10};

/*********************************************************/
/*  Main Program                                         */
/*********************************************************/
void main ( void )
{
   //Dialing
   dialing();          //DTMF dialing
    while (1)
   {
      //Write user program here
    }
}

/*********************************************************/
/*  Initialize Program                                   */
/*********************************************************/
//Initialize DTMF function
void init_DTMF(void)
{
   set_imask_ccr(1);                // Interrupt Disable
   init_AEC();
   #if (PWM_use==1)
   init_PWM1(0); //Select conversion period = 512/(PWM input clock)
   #else
   init_PWM2(0); //Select conversion period = 512/(PWM input clock)
   #endif
}

void init_PWM1(unsigned char selClk1)
{
   if (selClk1 <= 3)              // Check if valid, otherwise PWM2 is off
   {
      P_IO.PMR9.BIT.PWM1 = 1;      // Configure P91 as PWM2 output pin
      P_PWM1.PWCR1.BYTE = selClk1; // Clock select for PWM2,write only
   }
}

void init_PWM2(unsigned char selClk2)
{
   if (selClk2 <= 3)              // Check if valid, otherwise PWM2 is off
   {
      P_IO.PMR9.BIT.PWM2 = 1;      // Configure P91 as PWM2 output pin
      P_PWM2.PWCR2.BYTE = selClk2; // Clock select for PWM2,write only
   }
```

```
}
void off_DTMF(void)
{
   P_SYSCR.IENR2.BIT.IENEC = 0;
   //compiler directive to select which code to be compile
   #if (PWM_use==1)
   P_IO.PMR9.BIT.PWM1 = 0;         // Turn off PWM1
   #else
   P_IO.PMR9.BIT.PWM2 = 0;         // Turn off PWM2
   #endif
}

/*********************************************************/
/*  Initialize Program                                   */
/*********************************************************/
void warm_up(void)
{
   set_imask_ccr(0);               // Interrupts, 0-Enable, 1-Disable
   while(count<0x3000) ;
   set_imask_ccr(1);               // Interrupts, 0-Enable, 1-Disable
   Ready = 1;
}

/*********************************************************/
/*  DTMF dialing Program                                 */
/*********************************************************/
void dialing(void)
{
   init_DTMF();

   warm_up();

   i = 0;
   while (PHONE_NO[i]!=0x10)
   {
      inc1 = tone[PHONE_NO[i]][0];
      inc2 = tone[PHONE_NO[i]][1];

      //Dialing
      set_imask_ccr(0);                  // Interrupts, 0-Enable, 1-Disable
      for (j=0; j<15000; j++) ;          // short delay Tone dialing
      set_imask_ccr(1);                  // Interrupts, 0-Enable, 1-Disable

      storeCount(0);
      for (j=0; j<15000; j++) ;// short delay interval between 2 phone digit

      i++;                               //next digit
   }

   set_imask_ccr(1);                     // Interrupts, 0-Enable, 1-Disable

   off_DTMF();
```

```
}

/*********************************************************/
/*  Write each digital code into PWDR registers          */
/*********************************************************/
void storeCount(unsigned short PWDRval_2)
{
   //compiler directive to select which code to be compile
   #if (PWM_use==1)
   P_PWM1.PWDRL1.BYTE = (unsigned char)(PWDRval_2 & 0x00FF);
                                // Write lower 8bits of 10bits data
   P_PWM1.PWDRU1.BYTE = (unsigned char) ((PWDRval_2 & 0x0300) >> 8);
                                // Write upper 8bits of 10bits data
   #else
   P_PWM2.PWDRL2.BYTE = (unsigned char)(PWDRval_2 & 0x00FF);
                                // Write lower 8bits of 10bits data
   P_PWM2.PWDRU2.BYTE = (unsigned char) ((PWDRval_2 & 0x0300) >> 8);
                                // Write upper 8bits of 10bits data
   #endif
}

/*********************************************************/
/*  AEC Interrupt Service Routine                        */
/*********************************************************/
void aecint (void)
{
   P_SYSCR.IRR2.BIT.IRREC = 0;     // Clear IRREC flag

   if(P_AEC.ECCSR.BIT.OVL == 1)    // Check for ECL overflow flag
   {  P_AEC.ECCSR.BIT.OVL = 0;     // Clears flag

      if(Ready == 0)
      {
         storeCount(count++/128);
      }
      else
      {
         final = (Sine_Table[lowcnt])/2+(Sine_Table[hicnt])/2;
         storeCount(final);
         lowcnt = lowcnt + inc1;
         if(lowcnt>255) lowcnt = lowcnt-255;
                                // If reached end of 1 period, then reset
         hicnt = hicnt + inc2;
           if(hicnt>255) hicnt = hicnt-255;
                                // If reached end of 1 period, then reset
      }
   }
}

void init_AEC(void)
{
   P_AEC.ECCSR.BYTE = 0x15;
   P_AEC.ECCR.BYTE = 0x10;
```

```
    P_SYSCR.IRR2.BIT.IRREC = 0;     // Clear IRREC flag
    P_SYSCR.IENR2.BIT.IENEC = 1;    // AEC Interrupt Request
}
```

以下に“intprg.c”の割り込みサービスプログラムを示します。次のコードを挿入してください。

```
extern void aecint (void);              //insert AEC ISR function
.
.
.
.
.
.
__interrupt(vect=12) void INT_Counter(void)
{
aecint();                               //insert AEC ISR function
}
```

5. 参考文献

1. PWM Sine Wave Generation, (Application Note ref. no: AN0303003, http://sg.renesas.com,)
2. Use PWM as A DAC, (Application Note ref. no: AN0303003, http://sg.renesas.com,)

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	04.08.06	—	初版発行

安全設計に関するお願い

1. 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

本資料ご利用に際しての留意事項

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてルネサス テクノロジが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ルネサス テクノロジは責任を負いません。
3. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ(http://www.renesas.com)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したものですが万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任は負いません。
6. 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 本資料に関し詳細についてのお問い合わせ、その他お気付きの点がございましたらルネサス テクノロジ、ルネサス販売または特約店までご照会ください。