ADuC702xファミリーのPWMを使ったアナログ出力の発生

著者：Aude Richard

概要

ADuC702xファミリーは、32ビットARM7TDMIマイクロコントローラと高精度のアナログ・ブロックを内蔵しています。モデルに応じて、最大4個の12ビットDACが使用可能です。アプリケーションによっては、さらにアナログ出力を追加する必要がある場合があり、PWMを低分解能のDACとして使用することもできます。このアプリケーション・ノートでは、PWMから追加のアナログ出力を発生させる方法について説明します。

動作原理

代表的なPWM信号を図1に示します。基本周波数（スイッチング周波数）は固定で、パルス幅（デューティ・サイクル）を変えます。

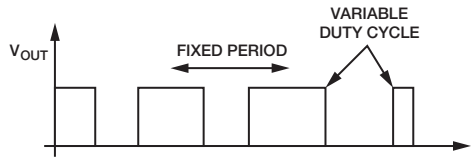

図1. 代表的なPWM波形

パルス幅が信号振幅に比例し、波形の周波数は一定です。

ADuC702xのPWMブロック

ADuC702xのPWMブロックは、3チャンネルのPWMで構成されています。これら3個のPWMは共通のスイッチング周波数を使い、PWMDAT0レジスタを使って343.99Hz～11.27MHzの範囲で設定することができます。

スイッチング周波数（f_{PWM}）は、次のように計算されます。

$$f_{PWM}=f_{CORE}/(2\times PWMDAT0)$$

スイッチング周波数は5.5kHzに設定します（PWMDAT0＝0x1000）。この設定で、86Hzの正弦波になります（正弦波当たり64サンプル）。

各チャンネルのデューティ・サイクルは独立であるため、3相を独立した3個のPWMとして使うことができます。このアプリケーション・ノートでは、PWMブロックの1チャンネルであるチャンネル0について説明します。

各PWMチャンネルは、ハイサイドとローサイドの出力を持っています。デューティ・サイクルは、ハイサイドでは50%～100%の範囲で、ローサイドでは50%～0%の範囲で、それぞれ設定することができます。チャンネル0のデューティ・サイクルは、PWMCH0レジスタを使って設定します。デューティ・サイクルは、同期割込み時に各PWM周期で変調することができます。

次の例では、P3.0のPWM0H出力を使って正弦波を発生します。ハイサイド出力では、設定可能なデューティ・サイクルが50%～100%に制限されますが、クロスオーバー・オプション（PWMEN MMR）を使うと、内部で0Lと0Hとの間のスイッチングが可能になるため、0～50%のデューティ・サイクルが可能になります。

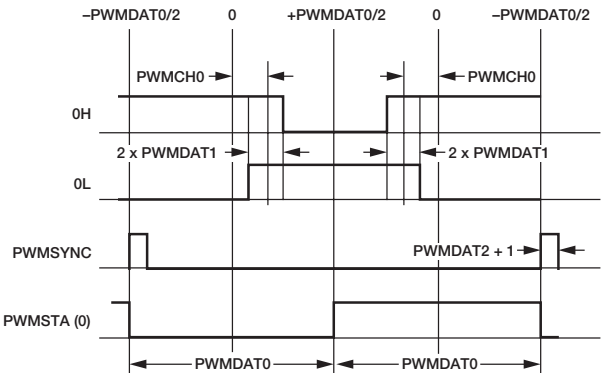

図2. シングル更新モードのタイミング

PWMAレジスタの値を計算する式は次のようになります。

ハイサイド:

$$T_{0HH}=PWMDAT0+2(PWMCH0-PWMDAT1)\times t_{CORE}$$

$$T_{0HL}=PWMDAT0-2(PWMCH0-PWMDAT1)\times t_{CORE}$$

すなわち、

$$d_{OH}=1/2+(PWMCH0-PWMDAT1)\,/\,PWMDAT0$$

デッドタイムを使わない場合は、

$$PWMCH0=(d_{OH}-1/2)\times PWMDAT0$$

ローサイド:

$$T_{0LH}=PWMDAT0-2(PWMCH0+PWMDAT1)\times t_{CORE}$$

$$T_{0LL}=PWMDAT0+2(PWMCH0+PWMDAT1)\times t_{CORE}$$

すなわち、

$$d_{OL}=1/2-(PWMCH0+PWMDAT1)\,/\,PWMDAT0$$

デッドタイムを使わない場合は、

$$PWMCH0=(1/2-d_{OL})\times PWMDAT0$$

REV. 0

アナログ・デバイセズ株式会社

本　　社／〒105-6891 東京都港区海岸1-16-1 ニューピア竹芝サウスタワービル
電話03(5402)8200

大阪営業所／〒532-0003 大阪府大阪市淀川区宮原3-5-36 新大阪MTビル2号
電話06(6350)6868

ハードウェアの考慮事項

PWM波形からアナログ信号への変換では、図3に示すようなローパス・フィルタが必要です。

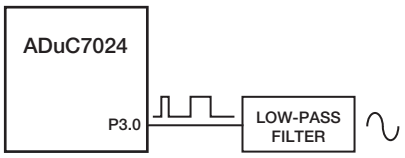

図3. 外付けフィルタ

AC信号に対して、シンプルな2段接続の2極RCフィルタを使って正弦波を再生することができます。この例でのスイッチング周波数は5.5kHzであり、必要とされる正弦波は86Hzであるため、フィルタの遮断周波数は約550Hzになります。この周波数は、PWMスイッチング周波数の1/10以下ですが、帯域幅の端から十分離れているので十分な減衰量を提供します。

フィルタのカットオフ周波数は、次のように計算されます。

$$FC = 1/2\pi RC$$

初段のフィルタはR＝2kΩとC＝0.2μF、2段目のフィルタはR＝1MΩとC＝200pFです。

AC信号出力の代わりにDCレベルが必要な場合は、ローパス・フィルタを使って発生することができます。値を保持するために外付け容量が必要になります。330kΩの抵抗と0.047μFの容量で、10Hzのカットオフ周波数が得られます。スイッチング周波数は5.5kHzを使います。

ソフトウェア

PWMCH0レジスタの範囲は[0;PWMDAT/2]とし、0は50%に対応します。表Iに、レジスタ値とデューティ・サイクルの対応を示します。

表I. MMR値とデューティ・サイクル値

High Side		Low Side	
PWMA	Duty Cycle	PWMA	Duty Cycle
0	50%	0	50%
PWMDAT0/2	100%	PWMDAT/2	0%

DC電圧

出力のフィルタで、小さい減衰が発生します。50%デューティ・サイクルで、1.25Vと測定されます。したがって、500mVの出力を得るときは、20%のデューティ・サイクルを設定する必要があります。50%より小さいデューティ・サイクルの式は、

$$PWMCH0 = PWMDAT0\ (1/2 - d_{OH})$$

500mVの出力を得るためには、PWMCH0＝0x999かつPWMDAT0＝0x2000とします。

正弦波

正弦波値は配列に保存されます。前半の正弦波はハイサイド（50%～100%）で出力し、後半はローサイドで出力する必要があります。

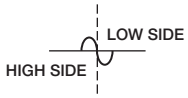

図4. 正弦波

アルゴリズムとしては、次のように表されます。

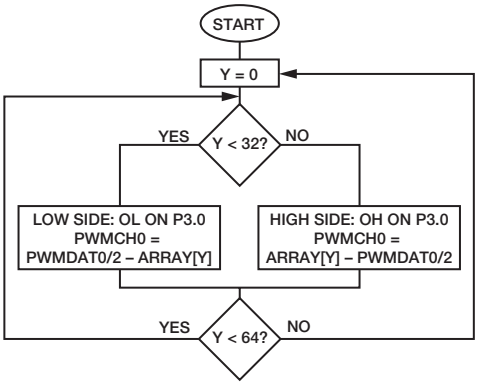

図5. フローチャート

関連コードについては付録Bを参照してください。

ADuC7020/ADuC7021/ADuC7022でのPWM

ADuC7020/ADuC7021/ADuC7022ではGPIO数が限られているため、PWMを使用できません。ただし、PLAを使って一度に1チャンネルの出力は可能です。関連コードについては付録Cを参照してください。

付録A：DC値用のソフトウェア

```
    GP3CON = 0x10000011;                    // Enable the PWM outputs to P3.0 and P3.1
                                            // Setup the PWM
    PWMCON = 0x0001;                        // 0x01 is enabled
    PWMDAT0 = 0x2000;                       // Period register
    PWMDAT1 = 0x00;                         // Dead time
    PWMDAT2 = 0xFF;                         // PWM pulse width
    PWMCFG = 0x00;                          // Chop
    PWMEN = 0x12F;                          // Enable low side on P3.0
    PWMA = 0x999;                           // Duty cycle of 20%
```

付録B：正弦波発生用のソフトウェア

```
volatile int y = 0 ;

const static unsigned short SinArray[64] = {
   0x07FF, 0x08C8, 0x098E, 0x0A51, 0x0B0F, 0x0BC4, 0x0C71, 0x0D12,
   0x0DA7, 0x0E2E, 0x0EA5, 0x0F0D, 0x0F63, 0x0FA6, 0x0FD7, 0x0FF5,
   0x0FFF, 0x0FF5, 0x0FD7, 0x0FA6, 0x0F63, 0x0F0D, 0x0EA5, 0x0E2E,
   0x0DA7, 0x0D12, 0x0C71, 0x0BC4, 0x0B0F, 0x0A51, 0x098E, 0x08C8,
   0x07FF, 0x0736, 0x0670, 0x05AD, 0x04EF, 0x043A, 0x038D, 0x02EC,
   0x0257, 0x01D0, 0x0159, 0x00F1, 0x009B, 0x0058, 0x0027, 0x0009,
   0x0000, 0x0009, 0x0027, 0x0058, 0x009B, 0x00F1, 0x0159, 0x01D0,
   0x0257, 0x02EC, 0x038D, 0x043A, 0x04EF, 0x05AD, 0x0670, 0x0736
   };

void initPWM(void){
   GP3CON = 0x10000011;                     // Enable the PWM outputs to the GPIO
   PWMCON = 0x0001;                         // 0x01 is enabled
   PWMDAT0 = 0x1000;                        // Period register
   PWMDAT1 = 0x00;                          // Dead time
   PWMDAT2 = 0xFF;                          // PWM pulse width
   PWMCFG = 0x00;                           // Chop
   PWMCH0 = 0x0000;                         // Channel 0
/* workaround for PWM_SYNC errata */
   PLAELM15 = 0x0035;                       // Configure individual elements
   PLAIRQ = 0x001F;                         // IRQ output configuration
   }

void Sinus_IRQ(){                           // Interrupt routine
   if((IRQSIG & PLA_IRQ0_BIT)!=0){          // Interrupt PWMSYNCH Signal workaround
     if (y<32) {                            // high side
       PWMEN = 0x02F;
       PWMDAT0 = 0x1000;
       PWMA = SinArray[y] - 0x800;
     }
     else {                                 // low side
       PWMEN = 0x12F;
       PWMDAT0 = 0x1000;
       PWMA = 0x800 - SinArray[y];
     }
     y++;
     if(y==64) y=0;
   }
   return ;
}
```

```
int main(void) {
  initPWM();
  IRQ = Sinus_IRQ;                          // Specify Interrupt Service Routine
  IRQEN = PLA_IRQ0_BIT ;                    // Enable PWMSYNCH IRQ
    while (1){}                             // Wait for PWMSYNC interrupt
  }
```

付録C：ADuC7020/ADuC7021/ADuC7022上でのPWM

```
// configuration of PLA, PWM and GPIO to output 16.384 kHz on P1.7

PWMCON = 0x1;                               // enables o/p of the pwm
GP3CON = 0x000000001;
PWMDAT0 = 0x055F;                           // PWM switching frequency of 16.384 kHz
PWMDAT1 = 0x0;                              // dead time is zero

// Configure Port Pins
GP1CON = 0x30000000;                        // If you want to drive the pwm onto
GP4CON = 0x30000000;                        // p1.7 you need at least element 15 as
                                            // it is the one feedback to Block0 Elt0

// PWM0 onto SPM7 via PLAO[0]
PLAELM0 = 0x0059;                           // PWM from element 15
PLAELM8 = 0x0035;                           // PWM input
PLAELM15 = 0x0059;                          // PWM from element
```

AN05567-0-6/05(0)-J