AVR123：AT90PWM81/161 A/D変換、対温度最適化

要点

- 温度に対するA/D変換結果の最適化
- A/D変換器に内部VREF使用時のATMEL® AT90PWM81/161に適用可

1. 序説

AT90PWM81/161は10ﾋﾞｯﾄ逐次比較A/D変換器(ADC)が特徴です。このADCは以下を提供する15ﾁｬﾈﾙのｱﾅﾛｸﾞ多重器に接続されています。

- 0V(GND)を参照する11点のｼﾝｸﾞﾙ ｴﾝﾄﾞ入力。
- A/D変換に先立って差動入力電圧で14dB(×5)、20dB(×10)、26dB(×20)、32dB(×40)の増幅段階を提供する設定可能な利得段とでなる2つの差動電圧入力の組み合わせ。増幅されたﾁｬﾈﾙに於いては8ﾋﾞｯﾄ分解能が期待できます。

この応用記述は温度に関してA/D変換結果を再調整する方法を説明します。

2. 動作の理屈 – A/D変換

2.1. VREF制御

A/D変換器(ADC)用の基準電圧(VREF)はADCに対する変換範囲を指示します。それは以下のどれかとして選択することができます。

- AVCC
- 内部2.56V基準電圧
- 外部AREFﾋﾟﾝに存在する電圧

2.56Vの内部基準VREFはﾊﾞﾝﾄﾞｷﾞｬｯﾌﾟ電圧から乗算後に生成されます。

2.2. VREF校正

この内部電圧基準は温度の関数です。

これは2.56V基準電圧の±1%の精度内で3Vと周辺温度の工場で校正されます。この校正の結果は識票列に格納されます。この最終検査"Amb.VREF"は(語)ｱﾄﾞﾚｽ$1Eに設定されます(ATMELの90PWM81/161ﾃﾞｰﾀｼｰﾄもご覧ください)。

尚、工場に於いてVREFﾚﾍﾞﾙの読み取りは105℃で成し遂げられます。この値も識票列に格納されます。この最終検査"Hot VREF"は(語)ｱﾄﾞﾚｽ$1Fに設定されます。

2.3. A/D変換

ｼﾝｸﾞﾙ ｴﾝﾄﾞ変換に関して、その変換結果は以下です。

ADC読み取り値=Vin×1023/VREF

ここで、Vinは選択した入力ﾋﾟﾝ上の電圧、VREFは選択した基準電圧です。

3. 補償方法

ATMEL AT90PWM81/161ﾏｲｸﾛ ｺﾝﾄﾛｰﾗは組み込み温度感知器を提供します。この特徴は電圧基準での温度変動の走行時補償に使用することができます。

3.1. VREF 対 温度

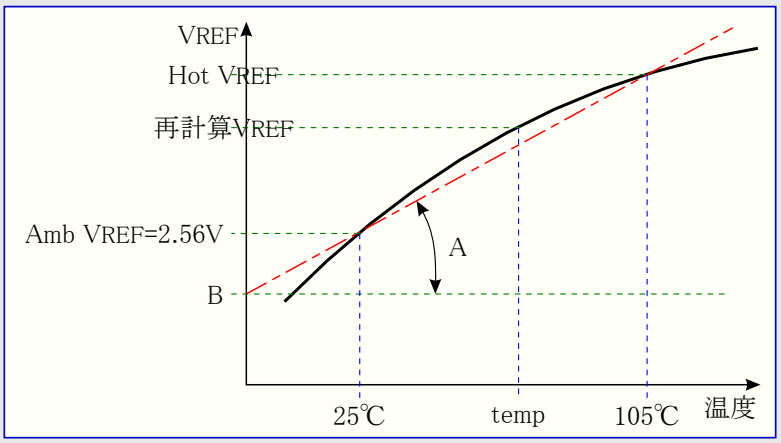
以下の例に於いて、VREF曲線の頂上を可能な最高温度に移動する電圧基準温度係数校正ﾚｼﾞｽﾀ(BGCRR)の既定形態設定を考察します。この形態設定ではより高い温度がより高いVREFになります。

8ﾋﾞｯﾄ ATMEL
ﾏｲｸﾛ ｺﾝﾄﾛｰﾗ

応用記述

Rev. 8270B-01/12, 8270BJ1-10/13

図3-1. VREF 対 温度



注:

1. Amb VREFは校正(±1%精度)に依存して厳密に2.56Vと等しい必要はありません。
2. Amb VREFは±1%精度、25℃での2.56Vに等しい工場で校正された値です。Hot VREFは工場に於いて105℃で読まれた値です。
3. Amb VREFとHot VREFは工場での検査操作中に識票列に格納され、ｿﾌﾄｳｪｱによって読むことができます。
4. 最終検査Amb VREFはｱﾄﾞﾚｽ$3D(上位ﾊﾞｲﾄ)と$3C(下位ﾊﾞｲﾄ)の2ﾊﾞｲﾄに設定されます。
5. 最終検査Hot VREF(読み込み専用)はｱﾄﾞﾚｽ$3F/$3Eの2ﾊﾞｲﾄに設定されます。

これらの定数はmVでの16進電圧値で、例えば$0A00は2560mVの16進値を表します。

3.2. 温度測定

この実装はAT90PWM81/161の組み込み温度感知器とで達成される測定を用います。
温度感知器が選択されているなら、温度測定式は次の通りです。

Temp(℃)=((([ADCH<<8|ADCL]-(273+25-TSOFFSET))×TSGAIN)/128)+25

TSGAINとTSOFFSETは工場での検査操作中に識票列に格納されます。
温度感知器変移(TSOFFSET)はｱﾄﾞﾚｽ$05の上位ﾊﾞｲﾄに設定されます。
ﾃﾞﾊﾞｲｽ温度感知器利得(TSGAIN)はｱﾄﾞﾚｽ$07の上位ﾊﾞｲﾄに設定されます(代表値は$80)。

3.3. VREF再計算

25℃と105℃間でVREF曲線対温度範囲は直線として推定することができます(図3-1.をご覧ください)。
VREF全体に渡る精度を改善するために、再計算VREFは以下として計算することができます。

再計算VREF=(A×Temp)+B

直線の既知の点は次の通りです。

Amb VREF = (A×(25℃))+B = 識票ﾊﾞｲﾄに格納されたﾃﾞｰﾀ
Hot VREF = (A×(105℃))+B = 識票ﾊﾞｲﾄに格納されたﾃﾞｰﾀ

AとBの変数はこれら2つの式から求めることができます。

3.4. A/D変換測定補償

A/D変換の測定結果は次式で補償することができます。

補償したA/D変換値=A/D変換読み込み値×再計算VREF/2.56

4. ﾊｰﾄﾞｳｪｱ形態設定

4.1. AT90PWM81/161

5つのﾃﾞﾊﾞｲｽが試験されました。それらのﾋｭｰｽﾞ形態設定は次の通りです。

- 拡張ﾋｭｰｽﾞ = $FD
- ﾋｭｰｽﾞ上位ﾊﾞｲﾄ = $D9
- ﾋｭｰｽﾞ下位ﾊﾞｲﾄ = $CC

4.2. STK521

クリスタル　　：8MHz
VREFの監視　　：AREF出力/VSS
UARTソフトウェア出力：RS232インターフェースに接続されたPB0

4.3. ハイパーターミナル

UARTビット速度　：19200bps

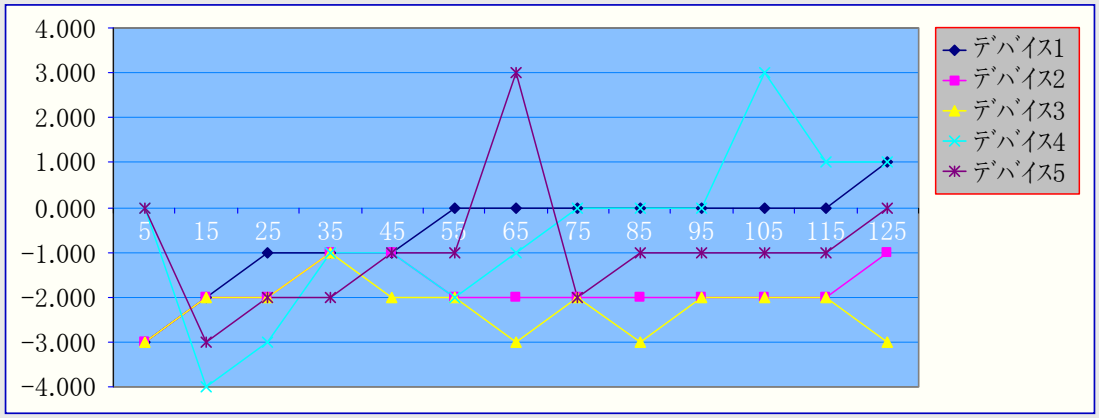
5. ソフトウェア形態設定

8章の(IARでコンパイルした)「コード例」をご覧ください。

6. 温度測定の結果

図6-1.の図は調べた5つのデバイスに関して温度測定の差が+4℃であることを確かにします。

図6-1. 温度測定の相違

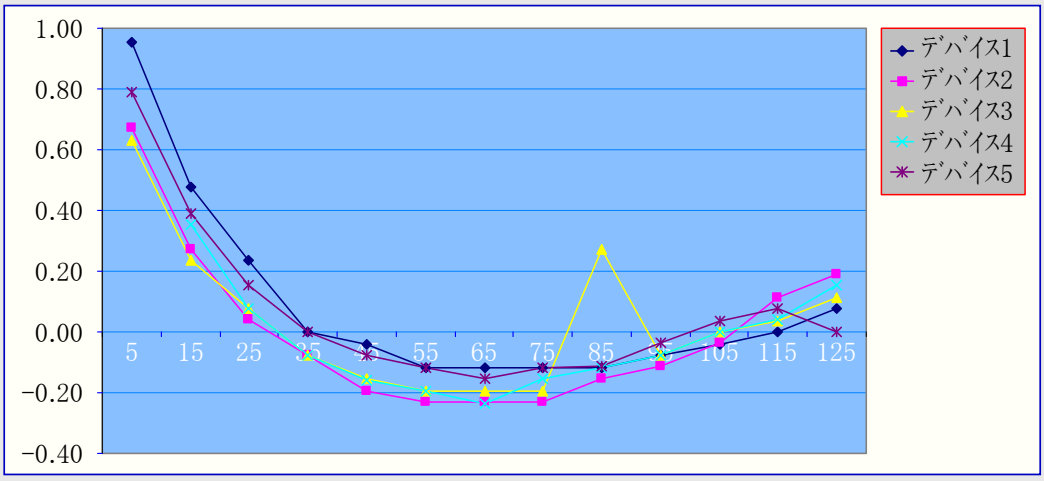


7. VREF再計算の結果

図7-1.の図は温度範囲に渡る実際のVREFに対する再計算したVREFの精度(%)を提供します。

精度=(再計算したVREF - VREF出力監視)/VREF出力監視

図7-1. 精度 対 温度



これらの代表的な結果は+5～+125℃の温度範囲に渡って再計算の精度が1%よりも良いことを確かにします。

8. ｺｰﾄﾞ例

8.1. C Main関数

```
//! //! Copyright (c) 2009 Atmel.
//! このﾌﾟﾛｸﾞﾗﾑは以下に対して繰り返しを使用します。
//! - 温度感知器の監視
//! - 実際内部VREFと等しくあるべきVREFの計算
/

//_____ ｲﾝｸﾙｰﾄﾞ ____________________________________________________________
#include "config.h"
#include "iopwm81.h"
#include "my_print.h"

//_____ 宣言 ____________________________________________________________

#define HIGHBYTE(v) ((unsigned char) (((unsigned int) (v)) >> 8));
#define LOWBYTE(v) ((unsigned char) (v));

int main(void)
{
unsigned char  gain, offset, temp, vref_amb_low, vref_hot_low,vref_amb_high, vref_hot_high,result;
unsigned int vref_recalc,vref_amb, vref_hot, n;
char  g,i;
float a,b;

  PORTB   = 0x00;
  DDRB    = 0xC7;
  ADCSRA |= 0x80;               /* ADEN=1 */
  ADMUX  |= 0x80;               /* VREF=2.56V */
  ADMUX  &= ~0x2F;
  ADMUX  |= 0x0C;               /* MUX=温度感知器 */
  ADCSRB  = 0x80;               /* ADC高速動作 + 自由走行(連続変換動作) */
  ADCSRA |= 0x04;               /* 16前置分周A/D変換ｸﾛｯｸ */
  ADCSRA |= (1<<ADSC);          /* 初回変換開始 */

  while (SPMEN==1);

  asm("LDI R17,$00");           /* 識票列内の温度感知器OFFSET読み込みLPM手順の始まり */
  asm("LDI R16,$05");
  asm("MOV R31,R17");           /*  */
  asm("MOV R30,R16");           /* ;Zﾎﾟｲﾝﾀ(R31=ZH R30=ZL)にｱﾄﾞﾚｽ設定 */
  SPMCSR=0x21;
  asm("LPM");                   /* ;識票列から温度感知器OFFSET値読み込み */
  asm("MOV R16, R0");           /* ;温度感知器OFFSET値をR16に複写 */
  asm("OUT 0x1B, R16");         /* ;温度感知器OFFSET値をGPIOR2に複写 */
  while (SPMEN==1);
  offset=GPIOR2;                /* 温度感知器OFFSET値戻り値設定 */

  asm("LDI R17,$00");           /* 識票列内の温度感知器GAIN読み込みLPM手順の始まり */
  asm("LDI R16,$07");
  asm("MOV R31,R17");           /*  */
  asm("MOV R30,R16");           /* ;Zﾎﾟｲﾝﾀ(R31=ZH R30=ZL)にｱﾄﾞﾚｽ設定 */
  SPMCSR=0x21;
  asm("LPM");                   /* ;識票列から温度感知器GAIN値読み込み */
  asm("MOV R16, R0");           /* ;温度感知器GAIN値をR16に複写 */
  asm("OUT 0x1A, R16");         /* ;温度感知器GAIN値をGPIOR1に複写 */
  while (SPMEN==1);
  gain=GPIOR1;                  /* 温度感知器GAIN値戻り値設定 */
```

```
asm("LDI R17,$00");             /* 識票列内のAmb.VREF(下位ﾊﾞｲﾄ)値読み込みLPM手順の始まり */
asm("LDI R16,$3C");
asm("MOV R31,R17");             /*  */
asm("MOV R30,R16");             /* ;Zﾎﾟｲﾝﾀ(R31=ZH R30=ZL)にｱﾄﾞﾚｽ設定 */
SPMCSR=0x21;
asm("LPM");                     /* ;識票列からAmb.VREF(下位ﾊﾞｲﾄ)値読み込み */
asm("MOV R16, R0");             /* ;Amb.VREF(下位ﾊﾞｲﾄ)値をR16に複写 */
asm("OUT 0x1A, R16");           /* ;Amb.VREF(下位ﾊﾞｲﾄ)値をGPIOR1に複写 */
while (SPMEN==1);
vref_amb_low=GPIOR1;            /* Amb.VREF(下位ﾊﾞｲﾄ)値戻り値設定 */

asm("LDI R17,$00");             /* 識票列内のAmb.VREF(上位ﾊﾞｲﾄ)値読み込みLPM手順の始まり */
asm("LDI R16,$3D");
asm("MOV R31,R17");             /*  */
asm("MOV R30,R16");             /* ;Zﾎﾟｲﾝﾀ(R31=ZH R30=ZL)にｱﾄﾞﾚｽ設定 */
SPMCSR=0x21;
asm("LPM");                     /* ;識票列からAmb.VREF(上位ﾊﾞｲﾄ)値読み込み */
asm("MOV R16, R0");             /* ;Amb.VREF(上位ﾊﾞｲﾄ)値をR16に複写 */
asm("OUT 0x1A, R16");           /* ;Amb.VREF(上位ﾊﾞｲﾄ)値をGPIOR1に複写 */
while (SPMEN==1);
vref_amb_high=GPIOR1;           /* Amb.VREF(上位ﾊﾞｲﾄ)値戻り値設定 */

asm("LDI R17,$00");             /* 識票列内のHot VREF(下位ﾊﾞｲﾄ)値読み込みLPM手順の始まり */
asm("LDI R16,$3E");
asm("MOV R31,R17");             /*  */
asm("MOV R30,R16");             /* ;Zﾎﾟｲﾝﾀ(R31=ZH R30=ZL)にｱﾄﾞﾚｽ設定 */
SPMCSR=0x21;
asm("LPM");                     /* ;識票列からHot VREF(下位ﾊﾞｲﾄ)値読み込み */
asm("MOV R16, R0");             /* ;Hot VREF(下位ﾊﾞｲﾄ)値をR16に複写 */
asm("OUT 0x1A, R16");           /* ;Hot VREF(下位ﾊﾞｲﾄ)値をGPIOR1に複写 */
while (SPMEN==1);
vref_hot_low=GPIOR1;            /* Hot VREF(下位ﾊﾞｲﾄ)値戻り値設定 */

asm("LDI R17,$00");             /* 識票列内のHot VREF(上位ﾊﾞｲﾄ)値読み込みLPM手順の始まり */
asm("LDI R16,$3F");
asm("MOV R31,R17");             /*  */
asm("MOV R30,R16");             /* ;Zﾎﾟｲﾝﾀ(R31=ZH R30=ZL)にｱﾄﾞﾚｽ設定 */
SPMCSR=0x21;
asm("LPM");                     /* ;識票列からHot VREF(上位ﾊﾞｲﾄ)値読み込み */
asm("MOV R16, R0");             /* ;Hot VREF(上位ﾊﾞｲﾄ)値をR16に複写 */
asm("OUT 0x1A, R16");           /* ;Hot VREF(上位ﾊﾞｲﾄ)値をGPIOR1に複写 */
while (SPMEN==1);
vref_hot_high=GPIOR1;           /* Hot VREF(上位ﾊﾞｲﾄ)値戻り値設定 */

vref_hot= (vref_hot_high *256) + vref_hot_low;
vref_amb = (vref_amb_high * 256) + vref_amb_low;
a=(vref_hot - vref_amb)*100;
a = a / 0x50;
b= vref_amb - ((a * 0x19)/100);
```

```
  while(1)
  {
    while (ADIF == 0);
    ADCSRA |=0x10;                                  /* ADIFﾘｾｯﾄ */
    temp =ADCL;
    result =(ADCH<<8);
    result = result | temp;
    result = result - (273+25-offset);
    temp = gain / 128;
    result = result * temp;
    temp = result +25;

    putchar(0x54);putchar(0x3D);print_hex(temp);    /* "T=..."16進形式で測定温度送信 */

    for(i=1;i<100;i++);   putchar(0x0D);  for(i=1;i<100;i++);  putchar(0x0A);

    vref_recalc = (((a * temp)/100) + b) ;          /* 測定温度に対して再計算されたVREFを取得 */

    putchar(0x41);putchar(0x3D);
    print_hex(vref_amb_high);
    print_hex(vref_amb_low);                        /* "A=..."16進形式でAmb.VREF送信 */
    for(i=1;i<100;i++);   putchar(0x0D);  for(i=1;i<100;i++);  putchar(0x0A);

    putchar(0x48);putchar(0x3D);
    print_hex(vref_hot_high);
    print_hex(vref_hot_low);                        /* "H=..."16進形式でHot VREF送信 */
    for(i=1;i<100;i++);   putchar(0x0D);  for(i=1;i<100;i++);  putchar(0x0A);

    putchar(0x52);putchar(0x3D);                    /* "R=..."16進形式で再計算VREF送信 */
    temp=HIGHBYTE(vref_recalc);
    print_hex (temp);
    temp=LOWBYTE(vref_recalc);
    print_hex (temp);                               /* "R=..."16進形式で再計算VREF送信 */
    for(i=1;i<100;i++);   putchar(0x0D);  for(i=1;i<100;i++);  putchar(0x0A);
    for(n=1;n<10000;n++)
    {
      for(i=1;i<100;i++);                           /* ﾊｲﾊﾟｰﾀｰﾐﾅﾙでの表示を改善するために遅延 */
    }
    ADMUX |=0x0C;
    ADCSRA |= (1<<ADSC);                            /* 温度感知器での新規変換開始 */
  }
}
```

8.2. Cソフトウェア UART関数

```
#include "my_print.h"
void print_hex(unsigned char n)
{
  unsigned char i;
  i=n>>4;
  if(i>9) putchar(i-0x0A+'A');
  else putchar(i+'0');
  i=n&0x0F;
  if(i>9) putchar(i-0x0A+'A');
  else putchar(i+'0');
}

void print_int(unsigned int n)
{
  print_hex(n>>8);
  print_hex(n);
}
```

8.3. ｱｾﾝﾌﾞﾘ言語 Soft_uart.s90

```
// 使用するAT90PWM81/161 MCU用のﾚｼﾞｽﾀ定義をｲﾝｸﾙｰﾄﾞ
#include   "iopwm81.h"

PUBLIC     putchar
PUBLIC     soft_uart_init

;***** ﾋﾟﾝ定義

TxD        EQU        0               ;送信ﾋﾟﾝはPDx

;***** 全体ﾚｼﾞｽﾀ変数

#define    bitcnt     R16             ;ﾋﾞｯﾄ計数器
#define    temp       R17             ;一時記憶ﾚｼﾞｽﾀ

#define    Txbyte     R18             ;送信されるべきﾃﾞｰﾀ

RSEG       CODE:CODE:NOROOT(1)

;***************************************************************************
;*
;* "putchar"
;*
;* このｻﾌﾞﾙｰﾁﾝは"Txbyte"ﾚｼﾞｽﾀに格納されたﾊﾞｲﾄを送信します。
;* 使用される停止ﾋﾞｯﾄ数はsb定数で設定されます。
;*
;* 語数              : RETを含めて14
;* 周期数            : ﾋﾞｯﾄ速度に依存
;* 下位側使用ﾚｼﾞｽﾀ   : なし
;* 上位側使用ﾚｼﾞｽﾀ   : 2 (bitcnt,Txbyte)
;* 使用ﾎﾟｲﾝﾀ         : なし
;*
;***************************************************************************
sb         EQU        1               ;停止ﾋﾞｯﾄ数(1,2,～)

putchar:   CLI                        ;全体割り込み禁止
;          LDI        bitcnt,(9+sb)   ;1+8+sb(sbは停止ﾋﾞｯﾄ数)
           MOV        Txbyte,R16
           LDI        bitcnt,(9+sb)   ;1+8+sb(sbは停止ﾋﾞｯﾄ数)
           COM        Txbyte          ;全ﾋﾞｯﾄ論理反転
           SEC                        ;開始ﾋﾞｯﾄ設定

putchar0:  BRCC       putchar1        ;対応ﾋﾞｯﾄ=0で分岐
           CBI        PORTB,TxD       ;対応ﾋﾞｯﾄ=1で'0'送出
           RJMP       putchar2        ;次へ

putchar1:  SBI        PORTB,TxD       ;対応ﾋﾞｯﾄ=0で'1'送出
           NOP

putchar2:  RCALL      UART_delay      ;1ﾋﾞｯﾄ時間遅延
           RCALL      UART_delay
           LSR        Txbyte          ;次ﾋﾞｯﾄをｷｬﾘｰﾌﾗｸﾞへ
           DEC        bitcnt          ;ﾋﾞｯﾄ計数器減数
           BRNE       putchar0        ;残ﾋﾞｯﾄ有りで次ﾋﾞｯﾄ処理へ
                                      ;残ﾋﾞｯﾄなしで、
           SEI                        ;全体割り込み再許可
           RET                        ;呼び出し元へ復帰
```

```
;***********************************************************************
;*
;* "UART_delay"
;*
;* この遅延ｻﾌﾞﾙｰﾁﾝはﾊﾞｲﾄを送受信する時にﾋﾞｯﾄ間で必要とされる遅延を生成します。
;* 総実行時間は定数"b"によって設定されます。
;*
;* 3×b+7周期 (RCALLとRETを含む)
;*
;* 語数              :RETを含めて4
;* 下位側使用ﾚｼﾞｽﾀ : なし
;* 上位側使用ﾚｼﾞｽﾀ :1(temp)
;* 使用ﾎﾟｲﾝﾀ         : なし
;*
;***********************************************************************

b               EQU     63

UART_delay:
;               LDI     temp,b
                NOP
                NOP
                LDI     R17,b
UART_delay1:    DEC     temp
                NRNE    UART_delay1
                RET

;***** ここからﾌﾟﾛｸﾞﾗﾑ実行開始
;***** 試験ﾌﾟﾛｸﾞﾗﾑ
soft_uart_init:
                SBI     PORTB,TxD       ;ﾎﾟｰﾄ ﾋﾟﾝ初期化
                SBI     DDRB,TxD
                RET

forever:        LDI     R18,0x55        ;'U'
                RCALL   putchar
                RJMP    forever

                END
```

9. 応用記述改訂履歴

頁番号は本資料への参照であることに注意してください。この章の改訂参照は資料の改訂に対してです。

9.1. 改訂8270A-09/10

1. 初版、資料2535I-04/08から作成

9.2. 改訂8270B-01/12

1. AT90PWM161を追加

10. 目次

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
TEL (+1)(408) 441-0311
FAX (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
TEL (+852) 2245-6100
FAX (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parking 4
D-85748 Garching b. Munich
GERMANY
TEL (+49) 89-31970-0
FAX (+49) 89-3194621

Atmel Japan
141-0032 東京都品川区
大崎1-6-4
新大崎勧業ﾋﾞﾙ 16F
ｱﾄﾒﾙ ｼﾞｬﾊﾟﾝ合同会社
TEL (+81)(3)-6417-0300
FAX (+81)(3)-6417-0370

本応用記述はATMELのAVR123応用記述(doc8270.pdf Rev.8270B-01/12)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には( )内に英語表記や略称などを残す形で表記しています。

青字の部分はﾘﾝｸとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。