

```

/*****
/* FILE      :SampleMtr.c
/* DATE      :Fri, Apr 06, 2001
/* DESCRIPTION :Main Program
*****/
#include "usbfunction.h" //USB関係の定義集です
#include "variable.h"//変数関係の定義集です

void main(void)
{
unsigned char i;
    DA.DADR2 = 0xff;//ポートのを出力に設定しています。
    Pd = 0x0000;//現在値を0に初期化しています。
}

void int1ms(void)
{ /* このルーチンでは位置制御(1ms割り込み)で処理するプログラムを記述します。
    割り込み時間は変更可能です。 */
    TPU0.TSR.BIT.TGFA = 0;//1ms割り込みフラグクリア
    /* 速度指令計算&速度リミットを計算します。 */
    if(Cms == 0x41){//位置制御ならば速度指令値計算
        Vd = (short)(((long)Kpp * ((long)Pd - (long)Pn)) >> 6);//速度指令値計算
        if(Vd > VIm) Vd = VIm;//速度リミット
        if(Vd < -1*VIm) Vd = -1*VIm;
    }
}

void int100us(void)
{ /* このルーチンでは速度電流制御(100us割り込み)で処理するプログラムを記述します。
    割り込み時間は変更可能です。 */
    TPU2.TSR.BIT.TGFA = 0;//100us割り込みフラグクリア
    RTCSR = 0x5A20;//ウォッチドッグタイマのカウンタをクリアしています。

    Pp = Pn;//エンコーダ値を保存します。
    Pn = TPU1.TCNT;//現在値読み

    if(PImH < Pn){//現在位置がリミットにひっかかっているか判定します。
        Pd = PImH;
        Arm |= 0x0001;//アラームセット
        servo(0);//サーボをオフします。
    }else if(PImL > Pn){
        Pd = PImL;
        Arm |= 0x0002;
        servo(0);
    }

    /* 速度計算 */
    if(Cms != 0x43){//電圧指令モードでなければ以下を実行します。
        Vn = (char)(Pn - Pp);//位置の差分から速度を計算します(100us間の変化パルス)
        Id = (short)((long)Kpv * ((long)Vd - (long)(Vn)) >> 4);//電流指令値計算
        if(Id > 1024)Id = 1024;//指令電圧計算がオーバーフローしないように制限します。
        if(Id < -1024)Id = -1024;
    }

    /* 指令電圧計算 */
    Vout = Kpi * Id;

    /* 回転方向 */
    if(Vout < 0){//計算されたVoutを絶対値化しています。
        Vout = abs(Vout);
        PORT.PEDR.BYTE &= ~0x80; //回転方向CCW(-)
    }else{
        PORT.PEDR.BYTE |= 0x80; //回転方向CW(+)
    }
};

```

```
if(Vout > 186){//指令できる値は最大186です。=PWMDUTY100%)
    Vout = 186;
}

Daout = (186 - Vout); //回路上の理由から、指令値を反転させます。

if(Cms == 0x43){//電圧指令モードであれば、設定された値を直接設定します。
    Daout = IFIX;
}

DA.DADR3 = (char)Daout; // 出力
}
```