

簡易電子メトロノーム

概要

メトロノーム(metronome)は、一定の間隔で音を刻み、ピアノやバイオリンなど、個人で楽器を演奏、あるいは練習する際に、テンポを合わせるために使う音楽用具です。もともとは機械式のものが主流でしたが、最近では電子式のものが多く市販されています。

そこで今回は、簡易な電子メトロノームを製作してみます。

<仕様>

- 拍子(BEAT)は、1~8拍子の範囲で設定可能とします。
- テンポ(TEMPO)は、1分間当たり40~210の範囲で設定可能とします。
- LEDと簡易サウンドで動作を表現します。

動作原理

<基準時間>

- PICのCCPモジュールをキャプチャモードで使用し、0.01秒の周期割り込みを発生させます。
- テンポが、最も遅い40であれば、150回($6000 \div 40$)の割り込みになります。
- テンポが、最も早い210であれば、28回($6000 \div 210$)の割り込みになります。

<簡易サウンド>

- 拍子の最初(小節の頭)では□1760Hz(高い“ド”)の周波数を100msec間出力します。
- 拍子の以降では□440Hz(低い“ド”)の周波数を100msec間出力します。
- mikroCが提供するサウンド関数を利用します。
 - Sound_Init関数 ポートとピンを指定してサウンドを初期化します。
 - Sound_Play関数 出力周波数と出力時間を指定し、サウンドを鳴らせます。

<拍子とテンポの記録>

- 拍子とテンポの値は、変更時に、EEPROMに書き込みます。
- 拍子とテンポの値は、起動時に、EEPROMより読み込みます。

<スイッチ入力>

- 設定スイッチ(拍子とテンポのアップダウン)の入力取りこぼしを防ぐ(レスポンスを早くする)ために、割り込み処理の中でスイッチの状態をチェックし、スイッチが押下されているかを判断します。

回路図



```
//*****
*
*/
    <電子メトロノーム>
*/
//*****
*
extern      void      main();
extern      void      metronome(int beat, int tempo);
extern      void      init_ccp_compare();
extern      void      interrupt();
extern      void      set_delay(int msec);
extern      void      delay();
extern      void      switch_check();
extern      void      do_mi_so_do();
//*****
*
//LCD
sbit      LCD_RS      at      RB2_bit;
sbit      LCD_EN      at      RB3_bit;
sbit      LCD_D7      at      RB7_bit;
sbit      LCD_D6      at      RB6_bit;
```

```

sbit    LCD_D5          at    RB5_bit;
sbit    LCD_D4          at    RB4_bit;
sbit    LCD_RS_Direction at    TRISB2_bit;
sbit    LCD_EN_Direction at    TRISB3_bit;
sbit    LCD_D7_Direction at    TRISB7_bit;
sbit    LCD_D6_Direction at    TRISB6_bit;
sbit    LCD_D5_Direction at    TRISB5_bit;
sbit    LCD_D4_Direction at    TRISB4_bit;
//SW
short   sw_flg = 0b00000000;
sbit    BEAT_UP         at    sw_flg.B0;
sbit    BEAT_DOWN       at    sw_flg.B1;
sbit    TEMPO_UP        at    sw_flg.B2;
sbit    TEMPO_DOWN      at    sw_flg.B3;
//LED
sbit    LED1            at    RA1_bit;
sbit    LED2            at    RA0_bit;
#define LED1_ON          LED1 = 1
#define LED1_OFF         LED1 = 0
#define LED2_ON          LED2 = 1
#define LED2_OFF         LED2 = 0
//*****
*
int      beat, tempo;
void     main()
{
    char      buf[6];
    //
    OSCCON = 0b01110000; //クロックを8MHzに設定します。
    ANSEL  = 0b00000000; //A/D変換モジュールは使用しません。
    TRISA  = 0b11111100;
    TRISB  = 0b00000000;
    //
    beat = EEPROM_Read(0) << 8;
    beat |= EEPROM_Read(1);
    tempo = EEPROM_Read(2) << 8;
    tempo |= EEPROM_Read(3);
    if ((beat < 1) || (beat > 8)) {
        beat = 4;
    }
    if ((tempo < 40) || (tempo > 210)) {
        tempo = 150;
    }
    //
    Sound_Init(&PORTB, 0);
    //
    Lcd_Init();
    Lcd_Cmd(_LCD_CLEAR);
    Lcd_Cmd(_LCD_CURSOR_OFF);
    Lcd_Out(1, 1, "metronome v1.00");
    do_mi_so_do();

```

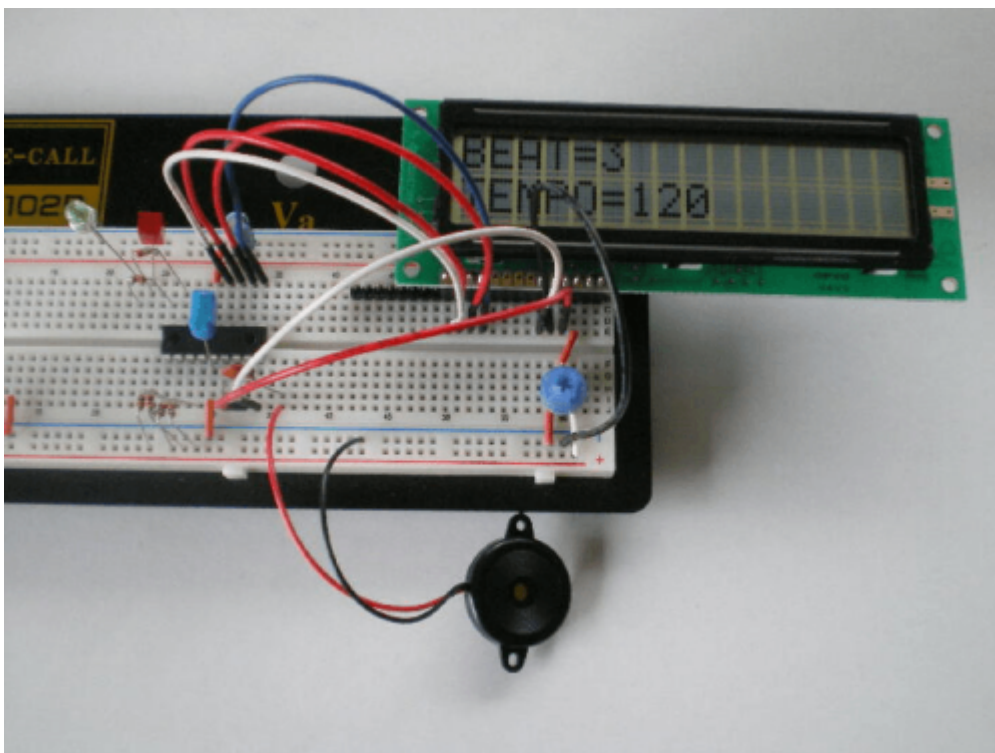
```
Lcd_Cmd(_LCD_CLEAR);
//
Lcd_Out(1, 1, "BEAT=");
WordToStr(beat, buf);
Lcd_Out(1, 6, &buf[4]);
Lcd_Out(2, 1, "TEMPO=");
WordToStr(tempo, buf);
Lcd_Out(2, 7, &buf[2]);
//
init_ccp_compare();
    // 割り込みを許可します。
INTCON.PEIE = 1;
INTCON.GIE = 1;
//
while (1) {
    metronome(beat, tempo);
}
}
//*****
*
void    metronome(int beat, int tempo)
{
    short    cnt;
    //
    set_delay(6000 / tempo);
    LED1_ON;
    Sound_Play(1760, 100);        // ド high
    LED1_OFF;
    delay();
    for(cnt = 0; cnt < (beat - 1); cnt++) {
        set_delay(6000 / tempo);
        LED2_ON;
        Sound_Play(440, 100);        // ド low
        LED2_OFF;
        delay();
    }
}
//*****
*
void    do_mi_so_do()
{
    Sound_Play(523, 250);        // ド (ピン)
    Sound_Play(659, 250);        // ミ (ボン)
    Sound_Play(784, 250);        // ソ (パン)
    Sound_Play(1047, 500);        // ド (ポーン)
    Delay_ms(500);
    Sound_Play(1047, 250);        // ド (ピン)
    Sound_Play(784, 250);        // ソ (ボン)
    Sound_Play(659, 250);        // ミ (パン)
    Sound_Play(523, 500);        // ド (ポーン)
```

```
    Delay_ms(1000);
}
//*****
*
void    switch_check()
{
    char        buf[6];
    //
    if (BEAT_UP) {
        if (beat < 8) {
            beat++;
        }
        WordToStr(beat, buf);
        Lcd_Out(1, 6, &buf[4]);
        EEPROM_Write(0, (beat >> 8) & 0xFF);
        EEPROM_Write(1, beat & 0xFF);
    }
    if (BEAT_DOWN) {
        if (beat > 1) {
            beat--;
        }
        WordToStr(beat, buf);
        Lcd_Out(1, 6, &buf[4]);
        EEPROM_Write(0, (beat >> 8) & 0xFF);
        EEPROM_Write(1, beat & 0xFF);
    }
    //
    if (TEMPO_UP) {
        if (tempo < 210) {
            tempo++;
        }
        WordToStr(tempo, buf);
        Lcd_Out(2, 7, &buf[2]);
        EEPROM_Write(2, (tempo >> 8) & 0xFF);
        EEPROM_Write(3, tempo & 0xFF);
    }
    if (TEMPO_DOWN) {
        if (tempo > 40) {
            tempo--;
        }
        WordToStr(tempo, buf);
        Lcd_Out(2, 7, &buf[2]);
        EEPROM_Write(2, (tempo >> 8) & 0xFF);
        EEPROM_Write(3, tempo & 0xFF);
    }
    sw_flg = 0b00000000;
    while ((PORTA & 0b00111100) != 0b00111100) {
        Delay_ms(100);
    }
}
//*****
```

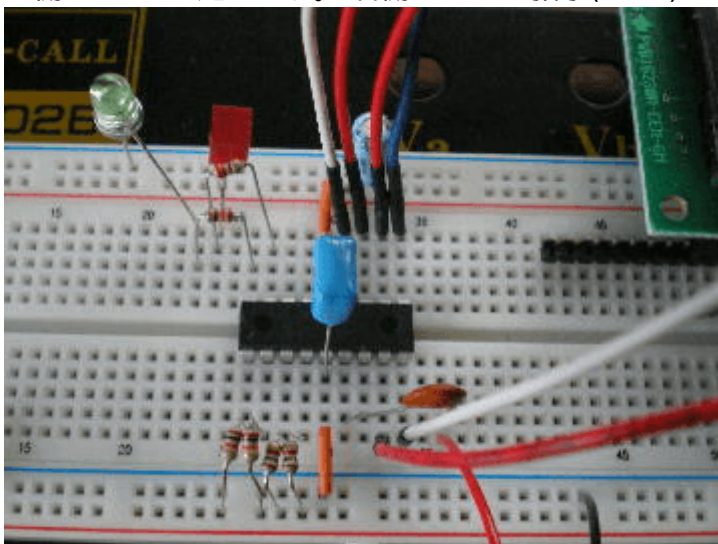
```
*
void    init_ccp_compare()
{
    // CCPの設定
    PIE1.CCP1IE = 1;
    PIR1.CCP1IF = 0;
    CCP1CON = 0b00001011;
    CCPR1L = 0xC4;           // 0.01sec...100hz...クロックが8Mhzの時
    CCPR1H = 0x09;           // 0.01sec...(1÷8000000)*4*8*2500
    // TIMER1の設定
    PIE1.TMR1IE = 0;
    PIR1.TMR1IF = 0;
    TMR1L = 0;
    TMR1H = 0;
    T1CON.T1CKPS0 = 1;
    T1CON.T1CKPS1 = 1;
    T1CON.TMR1ON = 1;
}
//*****
*
int      msec_cnt = 0;
void     interrupt()
{
    if (PIR1.CCP1IF == 1) {
        PIR1.CCP1IF = 0;
        //
        if (msec_cnt > 0) {
            msec_cnt--;
        }
        //
        if (PORTA.F2 == 0) {
            BEAT_UP = 1;
        }
        if (PORTA.F3 == 0) {
            BEAT_DOWN = 1;
        }
        if (PORTA.F4 == 0) {
            TEMPO_UP = 1;
        }
        if (PORTA.F5 == 0) {
            TEMPO_DOWN = 1;
        }
    }
}
//*****
*
void     set_delay(int msec)
{
    msec_cnt = msec;
}
```

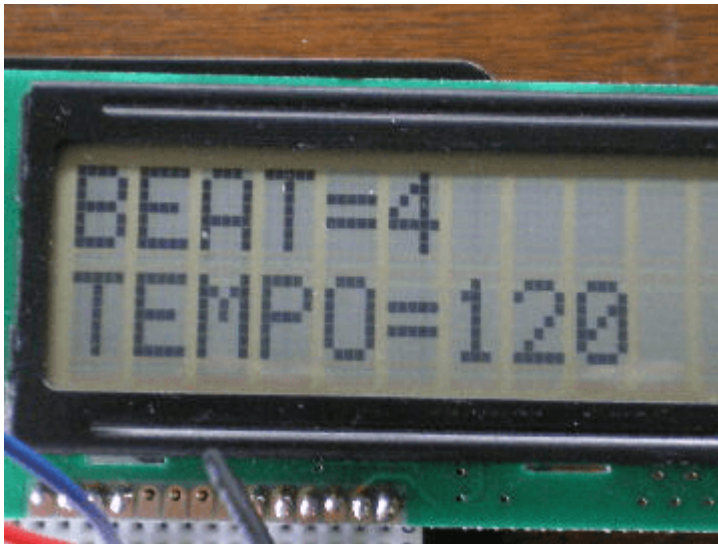
```
//*****  
*  
void    delay()  
{  
    while (msec_cnt != 0) {  
        switch_check();  
    }  
}  
}  
//*****  
*
```

動作確認



左側:PIC16F88廻りです。 右側:LCDへの拍子(BEAT)とテンポ(TEMPO)の表示内容です。





From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:135&rev=1588222087>

Last update: **2025/10/17 14:27**

