心拍検知ユニット

概要

以前から興味があった「心拍“心臓の鼓動”」を検知するユニットを作成しました。 今回は、単純に心拍を検知し、LEDとブザーで知らせるだけのものにしました。 将来的にはLCDに心拍数を表示させてみようと思っています。

動作原理

赤外線リモコンに使われる赤外線LEDと赤外線受光素子を使った方法を採用しました。 赤血球に含まれるヘモグロビンは赤外線を吸光する性質があり、動脈の脈拍動に伴って赤外線吸光度が変化することを捕捉して心拍を計測するものです。

赤外線LEDと赤外線受光素子には、フォトインタラプタを採用しました。 以前にジャンク袋を購入したときに10個程入っていたものです。 型番は無記入なのでよく分かりません。

- 赤外線受光素子の出力をLM386で200倍程度に増幅し、
- その後、ハイカットフィルタをとおして、
- 更に2Sk241で増幅し、
- その電圧をPIC12F683で処理し、
- LEDやブザーを鳴らせます。

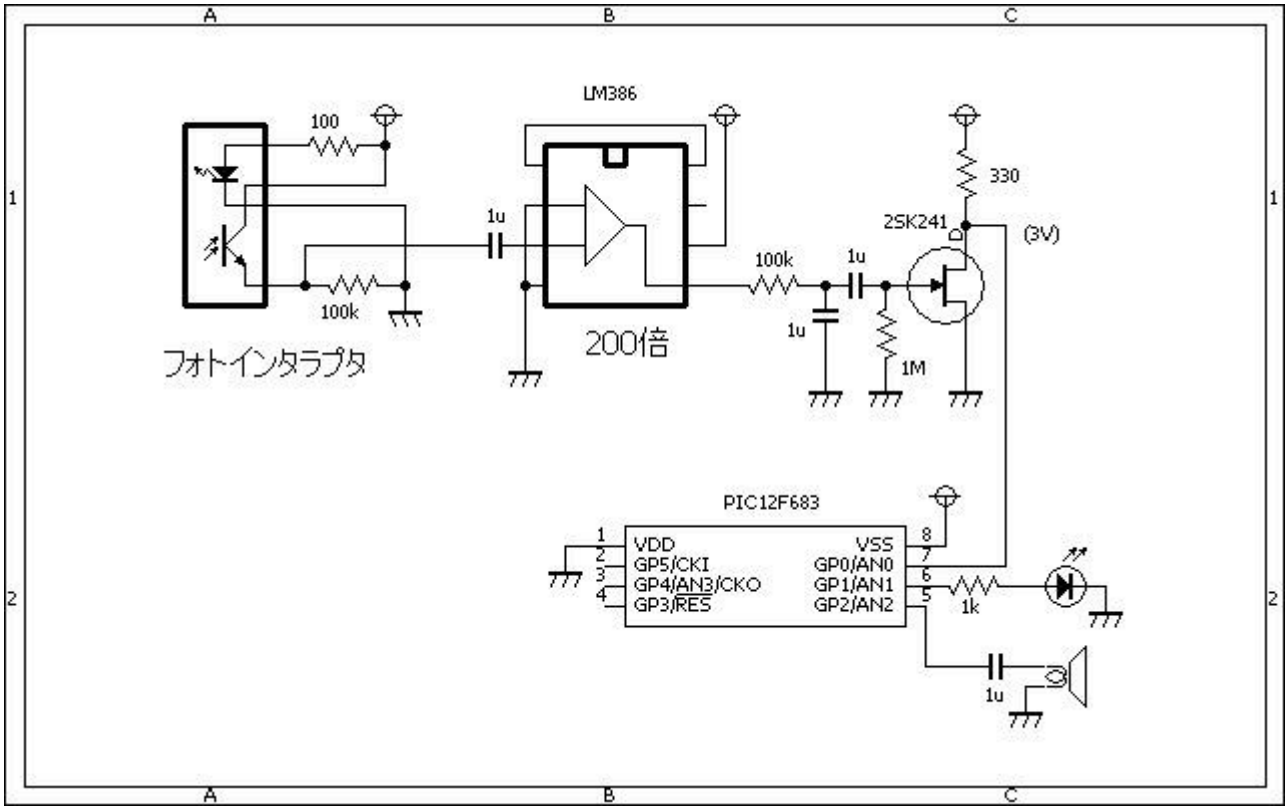
PICのソフトウエアでの処理は、割と単純です。 ＜旧版 FETのドレイン電圧（無信号時には約3V よりも、

- 高い電圧（3.1V以上）が連続して発生した場合
- 低い電圧（2.9V以下）が連続して発生した場合

で1波としてカウントし、LED,ブザーを制御します。 ＜改良版＞ 旧版は、基準電圧を絶対値の3Vとしましたが、精度を高めようとすると、電源電圧や2SK241のドレイン電圧（PICのA/D変換のリファレンス電圧そのものの精度を高めなければなりません。 それでは調整が面倒くさくなるので、基準電圧を相対値としました。

- 求め方は単純で、A/D変換の際に値を常に累積（約10秒分（A/D変換1万回分に相当）し、その平均値を基準とするものです。
  こうする事により、多少、電源電圧が変動したとしても正確に測定することが出来ます。

回路図



ソースコード

改良版のソースコード（絶対値比較から相対値比較へ変更しました。）

heartbeat2.c

```
//*****************************************************************************
*

void   Pwm_Change_DutyEx(unsigned int duty_ratio)
{
    CCPR1L = duty_ratio >> 2;
    CCP1CON.F6 = duty_ratio & 0b00000001;
    CCP1CON.F7 = (duty_ratio & 0b00000010) >> 1;
}

//*****************************************************************************
*
// 1msec毎に10回A/D変換し、その平均値を返す。
// 10sec毎の平均値（1万回分）を求める。

unsigned   int        ave, accCnt;
unsigned   long    acc;

unsigned   int    Adc_Read_Ex(unsigned short channel)
{
    unsigned    int        ad0, ad;
    unsigned    char    cnt;
```

```
    ad0 = 0;
    GPIO.F4 = 1;
    for (cnt = 0; cnt < 10; cnt++) {
        ad = Adc_Read(channel);
        ad0 += ad;
        //平均値を求める。
        acc += ad;
        accCnt++;
        if (accCnt == 10000) {
            ave = acc / 10000;
            accCnt = 0;
            acc = 0;
        }
        //
        Delay_us(1000);
    }
    GPIO.F4 = 0;
    return (ad0 / 10);
}

//*****************************************************************************
*

void   main()
{
    unsigned    int         ad0, old;
    unsigned    char    cnt;
    //
    OSCCON = 0b01110000;     //クロックは8Mhz
    CMCON0 = 0b00000111;     //コンパレータは使用しない。
    ANSEL = 0b00000001;         // A/D変換を使用する。
    TRISIO = 0b00000001;
    GPIO = 0b00001000;
    Pwm_Init(3000);         //発振周波数は、3kHz
    Pwm_Change_DutyEx((PR2 * 4) / 2);     //デューティ比は、50%
    Pwm_Stop();
    //
    acc = 0;
    accCnt = 0;
    ave = 3000 / 5;
    //
    while(1) {
        //心拍の立ち下がりをチェックする。
        cnt = 0;
        while(1) {
            ad0 = Adc_Read_Ex(0);
            if (ad0 > (ave + 20)) {     //平均値□0.1V以上かをチェックする。
                cnt++;
            } else {
                cnt = 0;
            }
```

```
            if (cnt > 5)     //5回連続するかをチェックする。
                break;
        }
        // ブザーを鳴らす。
        Pwm_Start();
        GPIO.F1 = 1;
        Delay_ms(10);
        GPIO.F1 = 0;
        Pwm_Stop();
        //心拍の立ち上がりをチェックする。
        cnt = 0;
        while(1) {
            ad0 = Adc_Read_Ex(0);
            if (ad0 < (ave - 20)) {     //平均値□0.1V以下かをチェックする。
                cnt++;
            } else {
                cnt = 0;
            }
            if (cnt > 5)     //5回連続するかをチェックする。
                break;
        }
    }
}

//*********************************************************************
*
```

旧版のソースコード

heartbeat.c

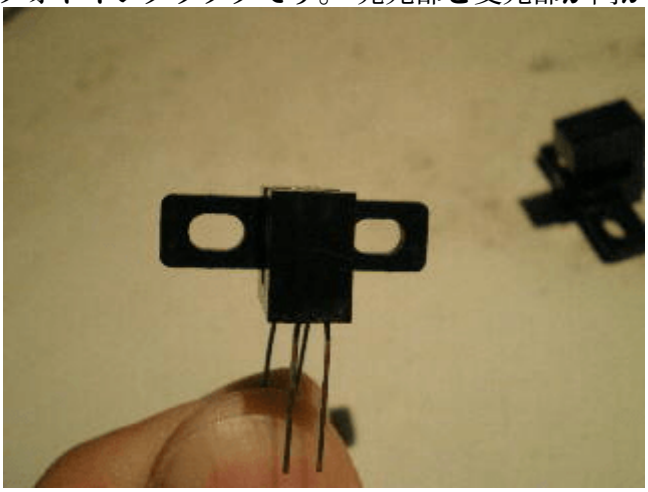
```
//*********************************************************************
*

void   Pwm_Change_DutyEx(unsigned int duty_ratio)
{
    CCPR1L = duty_ratio >> 2;
    CCP1CON.F6 = duty_ratio & 0b00000001;
    CCP1CON.F7 = (duty_ratio & 0b00000010) >> 1;
}

//*********************************************************************
*
//10回A/D変換し、その平均値を返す。クロックが8Mhzの場合、約1msec要する。
unsigned   int    Adc_Read_Ex(unsigned short channel)
{
    unsigned    int        ad0;
    unsigned    char    cnt;
    ad0 = 0;
    GPIO.F4 = 1;
```

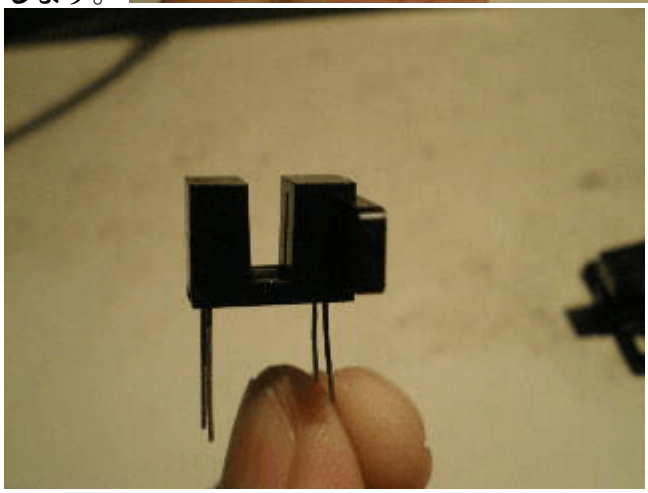
```
    for (cnt = 0; cnt < 10; cnt++) {
        ad0 += Adc_Read(channel);
    }
    GPIO.F4 = 0;
    return (ad0 / 10);
}

//*****************************************************************************
*

void   main()
{
    unsigned    int         ad0, old;
    unsigned    char    cnt;
    //
    OSCCON = 0b01110000;     //クロックは8Mhz
    CMCON0 = 0b00000111;     //コンパレータは使用しない。
    ANSEL = 0b00000001;         // A/D変換を使用する。
    TRISIO = 0b00000001;
    GPIO = 0b00001000;
    Pwm_Init(3000);
    Pwm_Change_DutyEx((PR2 * 4) / 2);
    Pwm_Stop();
    while(1) {
        //心拍の立ち下がりをチェックする。
        cnt = 0;
        while(1) {
            ad0 = Adc_Read_Ex(0);
            if (ad0 > (3100 / 5)) {    // 3.1V以上かをチェックする。
                cnt++;
            } else {
                cnt = 0;
            }
            if (cnt > 5)     //5回連続するかをチェックする。
                break;
        }
        //ブザーを鳴らす。
        Pwm_Start();
        GPIO.F1 = 1;
        Delay_ms(10);
        GPIO.F1 = 0;
        Pwm_Stop();
        //心拍の立ち上がりをチェックする。
        cnt = 0;
        while(1) {
            ad0 = Adc_Read_Ex(0);
            if (ad0 < (2900 / 5)) {    // 2.9V以下かをチェックする。
                cnt++;
            } else {
                cnt = 0;
            }
```

```
            if (cnt > 5)     //5回連続するかをチェックする。
                break;
        }
    }
}

//*****************************************************************************
*
```

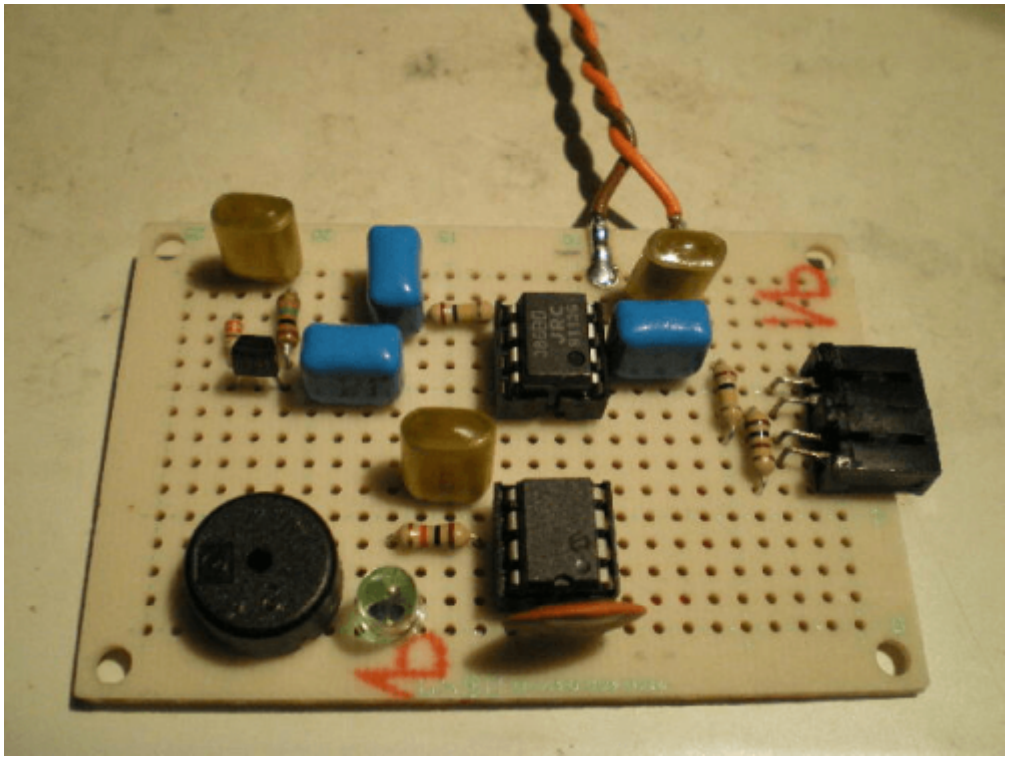
動作確認

使用したフォトインタラプタです。 発光部と受光部が向かい合わせなので、ニッパーで切り離して使用します。

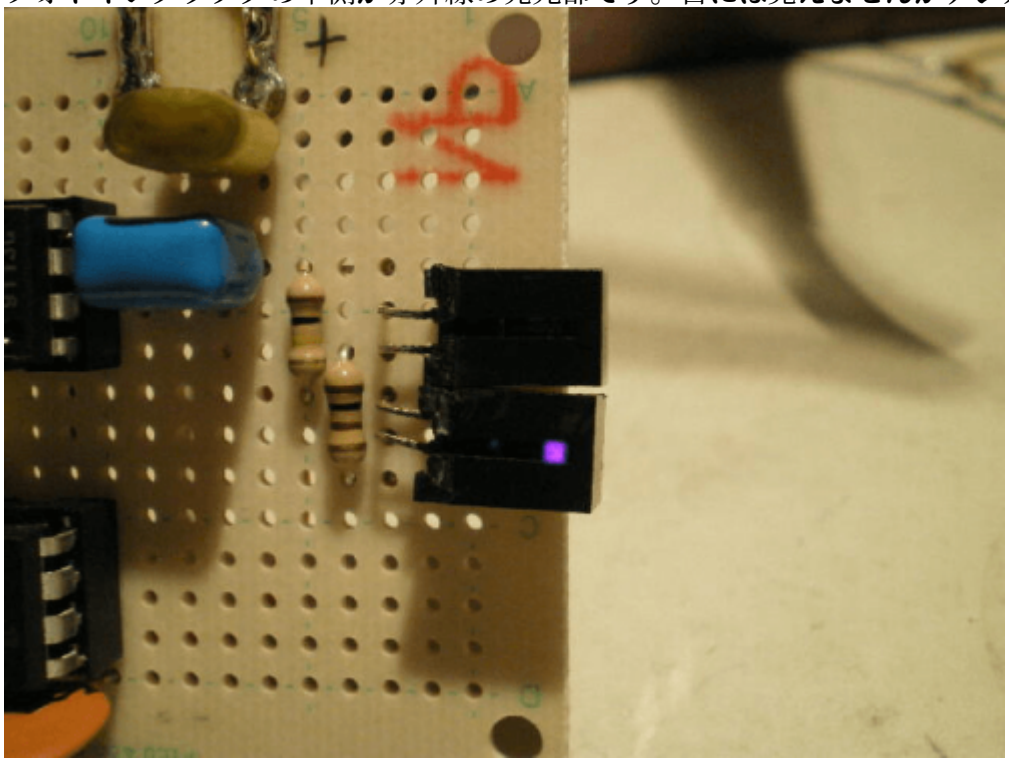




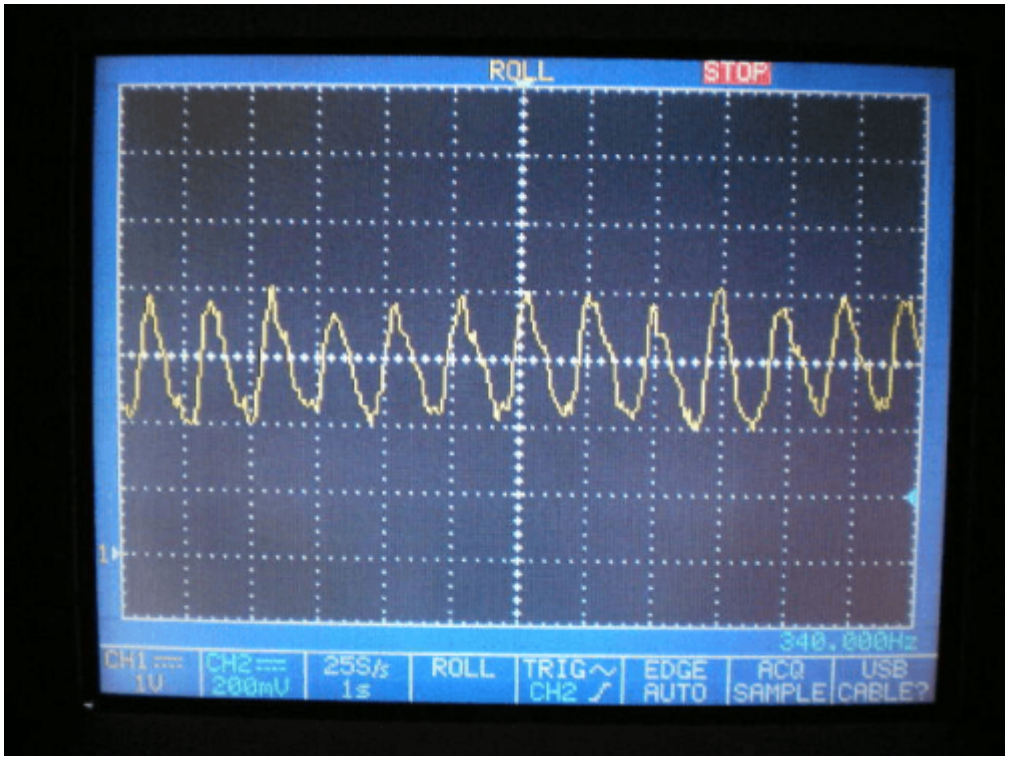
全体の回路です。右側の黒い部品がフォトインタラプタです。 この上に指をのせるだけです。心拍に合わせて左下のミニスピーカとLEDを制御します。



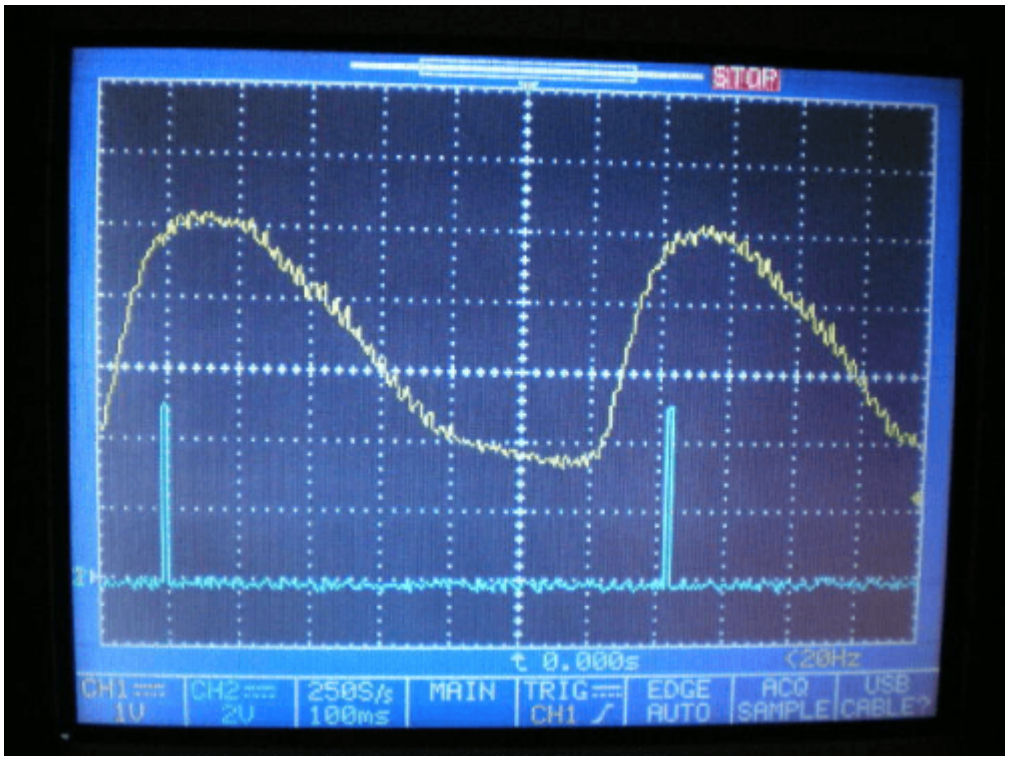
フォトインタラプタの下側が赤外線の発光部です。目には見えませんがデジカメには写りますね。



デジタルオシロで波形を観測しました。アナログオシロでは遅すぎてデジカメで写すことが出来ません。
レンジは、1v/DIV□1sec/DIVです□2SK241のドレイン電圧が無信号時には約3Vです。



ドレインの波形（つまり心拍の波形）とLEDの波形（つまり検知した波形）です。



如何ですか？ 自分の心拍を常時計測し健康維持に役立てることが出来ないものでしょうか？ 😄