タッチ式電鍵

概要

アマチュア無線では音声による交信の他に、電鍵を使ったモールス符号(トン・ツー)で交信を行う、モールス通信(CW)があります。

モールス通信を行う際には、無線機+電鍵を使用します。 電鍵には、ストレートキー(縦振り電鍵)、エレキー、バグキー等いろいろなタイプのものが販売されています。 しかしこれらは何らかの機械的な部分が存在するため、接点不良を発生する可能性があります。


ストレートキー(例)

今回は、一切、機械的部分の無い、人の指で軽く触れるだけで操作可能な「タッチ式電鍵」を製作してみました。

<仕様>

- 操作は、金属部分(1円玉位の大きさ)に指で触れるだけとします。
- 金属部分に指で触れるとLEDが点灯し、圧電スピーカからブザー音(約1kHz)が出ます。
- 無線機のCW端子の制御は、トランジスタ(オープンコレクタ)を使用します。
- 電源は、電池2本(3V)で動作可能とします。
- ブザー音が出るのでモールス練習機として使用すること出来ます。

動作原理

静電容量の変化により、タッチの有無を判別し、無線機のCW端子のオン・オフ制御を行います。

動作原理(ハードウェア)

◎静電容量検出

- センサー部分は、1円玉位の大きさの金属(銅板など)を使用します。
- 詳細については、簡易静電容量スイッチを参照してください。

◎CW端子制御

- 無線機のCW端子をオン・オフさせるためにトランジスタ(オープンコレクタ)を使用します。

動作原理(ソフトウェア)

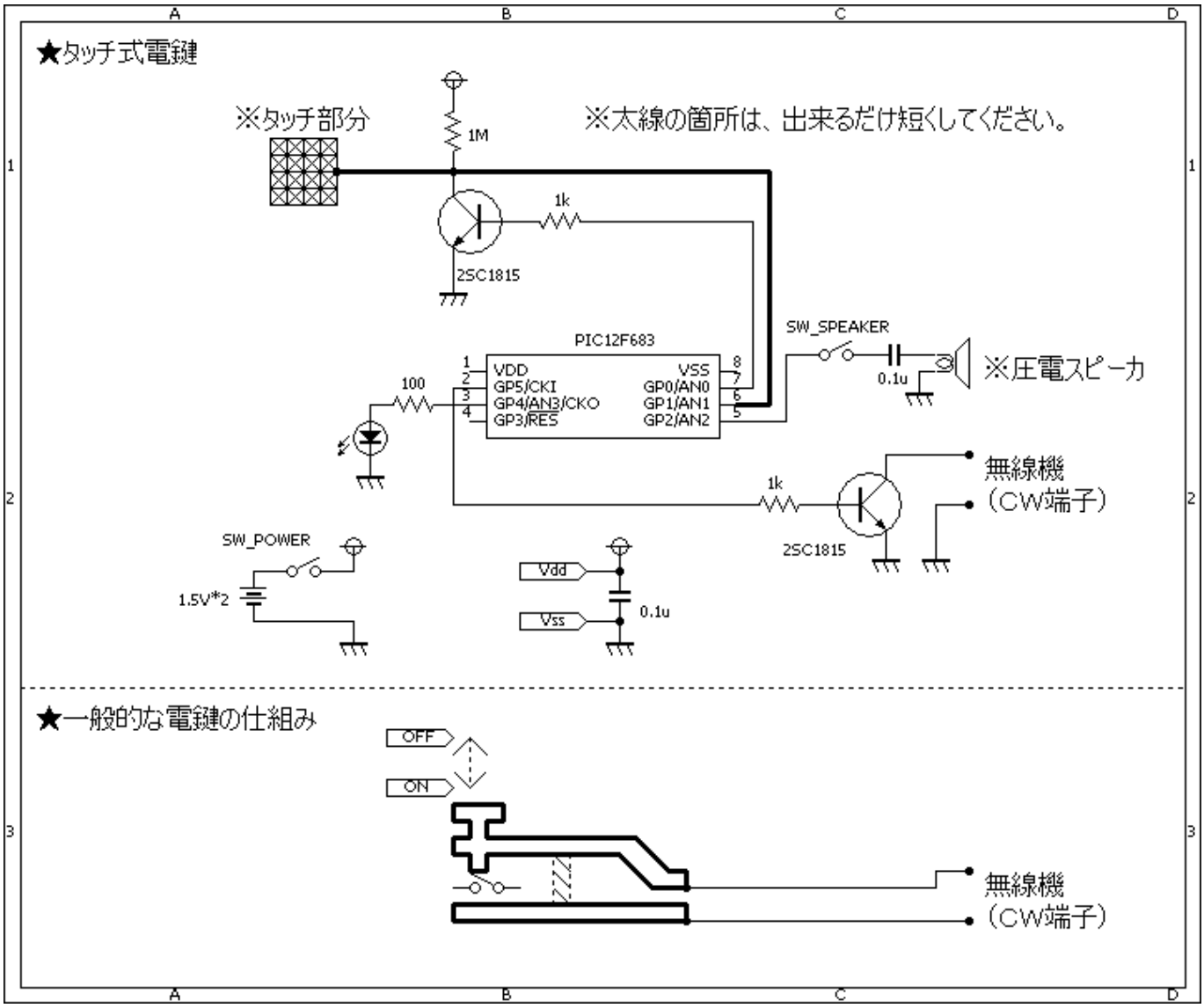
◎メイン処理(main)

- PWMを初期化します。
- コンパレータ参照電圧をオンにします。
- LEDを3回点滅させます。
- 静電容量を100回測定し平均値を求め、これを基準値(tm1)とします。
- LEDを3回点滅させます。
- 静電容量を測定します(tm2)
- tm1とtm2を比較し、20%以上変動していればタッチされたとみなします。
- タッチされていればLEDを点灯し、圧電スピーカからブザー音を出し、出力トランジスタ(CW用)をオンします。

◎静電容量測定処理(measurement)

- 詳細については、簡易静電容量スイッチを参照してください。

回路図



ソースコード

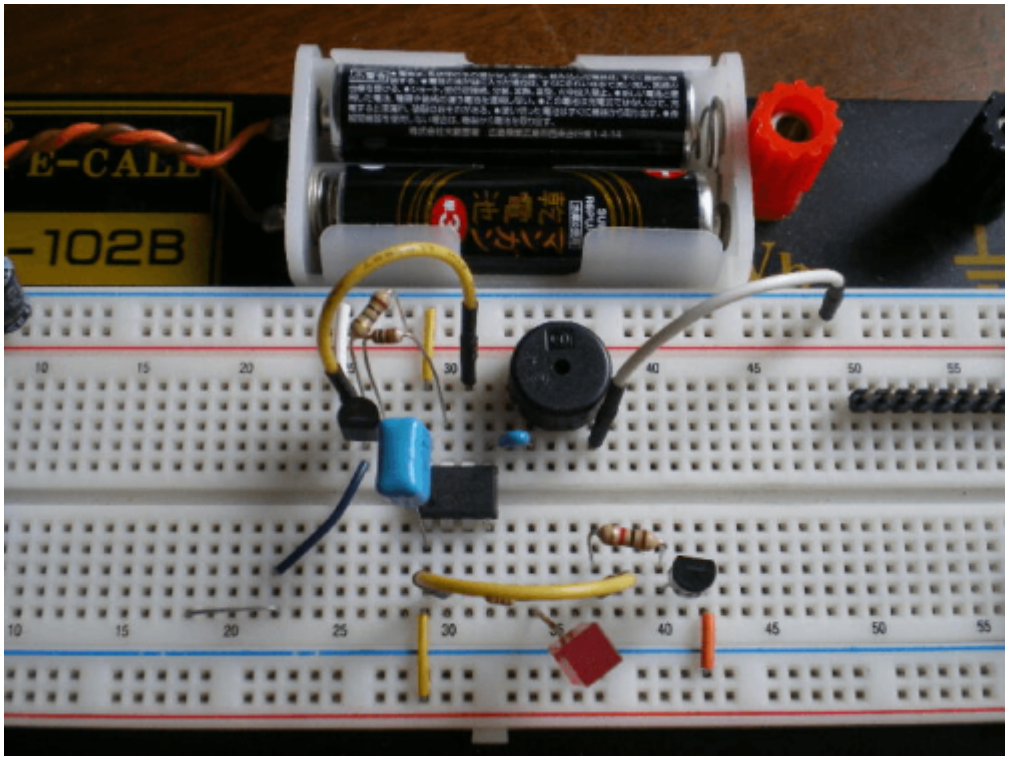
telegraph_key_v1_00.c

```
//*********************************************************************
*
/*
    <タッチ式電鍵（静電容量方式）>
*/
//*********************************************************************
*
//■■■マクロ定義■■■
#define BYTE    unsigned  char
#define WORD    unsigned  int
#define DWORD   unsigned  long
//
#define KEY     GPIO.B5
#define LED     GPIO.B4
#define SW      GPIO.B3
//*********************************************************************
*
extern  void    main();
extern  void    PWM1_Change_DutyEx(WORD duty_ratio);
extern  WORD    measurement();
extern  WORD    average(BYTE cnt);
//*********************************************************************
*
void    main()
{
        long    tm1, tm2;
        BYTE    cnt, on_cnt;
        //
        OSCCON = 0b01110000;        //クロックは8Mhz
        CMCON0 = 0b00000100;        //コンパレータを使用する。
        ANSEL  = 0b00000000;        // □□□変換は使用しない。
        TRISIO = 0b00001010;
        GPIO   = 0b00000000;
        //
        KEY = 0;
        //
        T1CON.T1CKPS1 = 0;
        T1CON.T1CKPS0 = 0;
        //
        PWM1_Init(1000);
        PWM1_Change_DutyEx((PR2 * 4) / 2);
        PWM1_Stop();
        //
        VRCON.VREN = 1;
        VRCON.VRR = 1;
        //
```

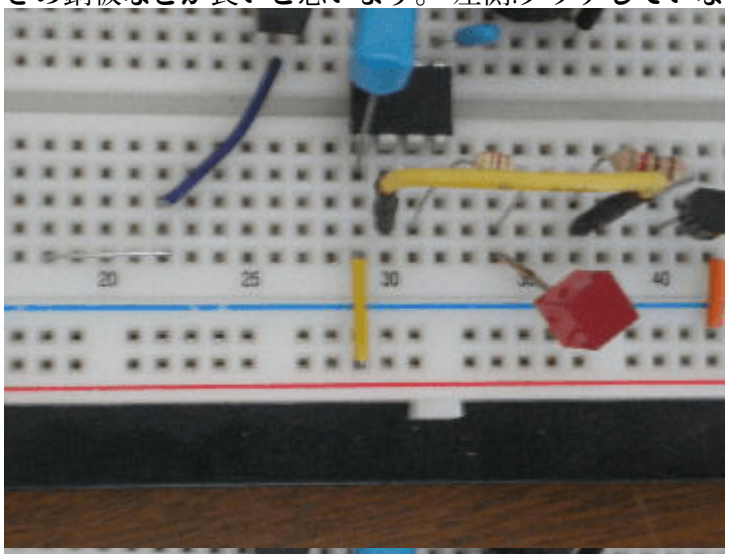
```
        for (cnt = 0; cnt < 3; cnt++) {
                LED = 1;
                Delay_ms(100);
                LED = 0;
                Delay_ms(100);
        }
        tm1 = Average(100);
        Delay_ms(500);
        //
        for (cnt = 0; cnt < 3; cnt++) {
                LED = 1;
                Delay_ms(100);
                LED = 0;
                Delay_ms(100);
        }
        //
        while (1) {
                on_cnt = 0;
                for (cnt = 0; cnt < 10; cnt++) {
                        tm2 = average(5);
                        if (labs(tm1 - tm2) > (tm1 / 5)) {
                                on_cnt++;
                        }
                }
                if (on_cnt > 5) {
                        LED = 1;
                        KEY = 1;
                        PWM1_Start();
                } else {
                        LED = 0;
                        KEY = 0;
                        PWM1_Stop();
                }
        }
}
//*****************************************************************************
*
WORD    measurement()
{
        WORD    tm;
        //
        TMR1L = 0;
        TMR1H = 0;
        PIR1.TMR1IF = 0;
        T1CON.TMR1ON = 0;
        //
        VRCON.VR3 = 0;
        VRCON.VR2 = 0;
        VRCON.VR1 = 0;
        VRCON.VR0 = 1;
        //
```

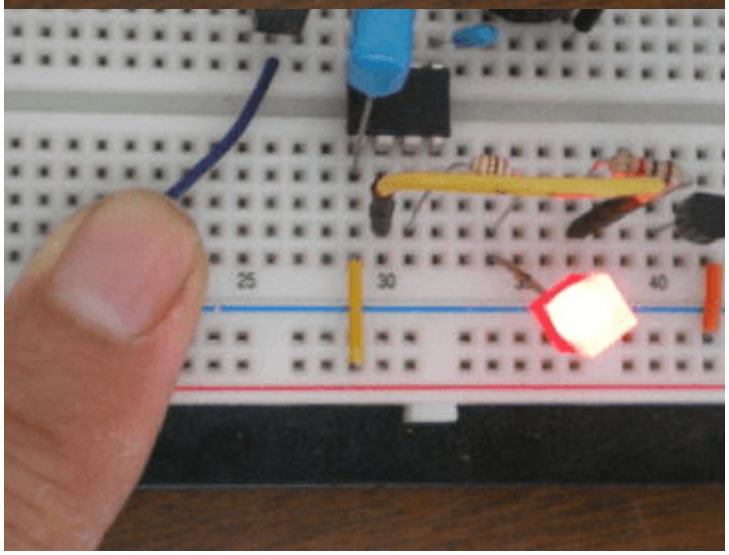
```
        GPIO.B0 = 1;
        Delay_us(100);
        GPIO.B0 = 0;
        //
        while (CMCON0.COUT == 1)
                ;
        //
        VRCON.VR3 = 1;
        VRCON.VR2 = 1;
        VRCON.VR1 = 1;
        VRCON.VR0 = 1;
        //
        T1CON.TMR1ON = 1;
        while (CMCON0.COUT == 1)
                ;
        //
        T1CON.TMR1ON = 0;
        tm = TMR1H << 8;
        tm |= TMR1L;
        //
        return(tm);
}
//*********************************************************************
*
WORD    average(BYTE cnt)
{
        DWORD   tm;
        BYTE    i;
        //
        tm = 0;
        for (i = 0; i < cnt; i++) {
                tm += measurement();
        }
        return(tm / cnt);
}
//*********************************************************************
*
void    PWM1_Change_DutyEx(WORD duty_ratio)
{
        CCPR1L = duty_ratio >> 2;
        CCP1CON.DC1B0 = duty_ratio & 0b00000001;
        CCP1CON.DC1B1 = (duty_ratio & 0b00000010) >> 1;
}
//*********************************************************************
*
```

動作確認



タッチセンサーには数cmの針金を使用しています。 実際に電鍵として使用する際には、1円玉位の大きさの銅板などが良いと思います。 左側:タッチしていない状態です。 右側:タッチした状態です。





如何ですか? これを2セット使うことにより、「タッチ式エレキー」を容易に製作することが出来そうですね。