

CWメモリーキーヤ(memory-keyer)

概要

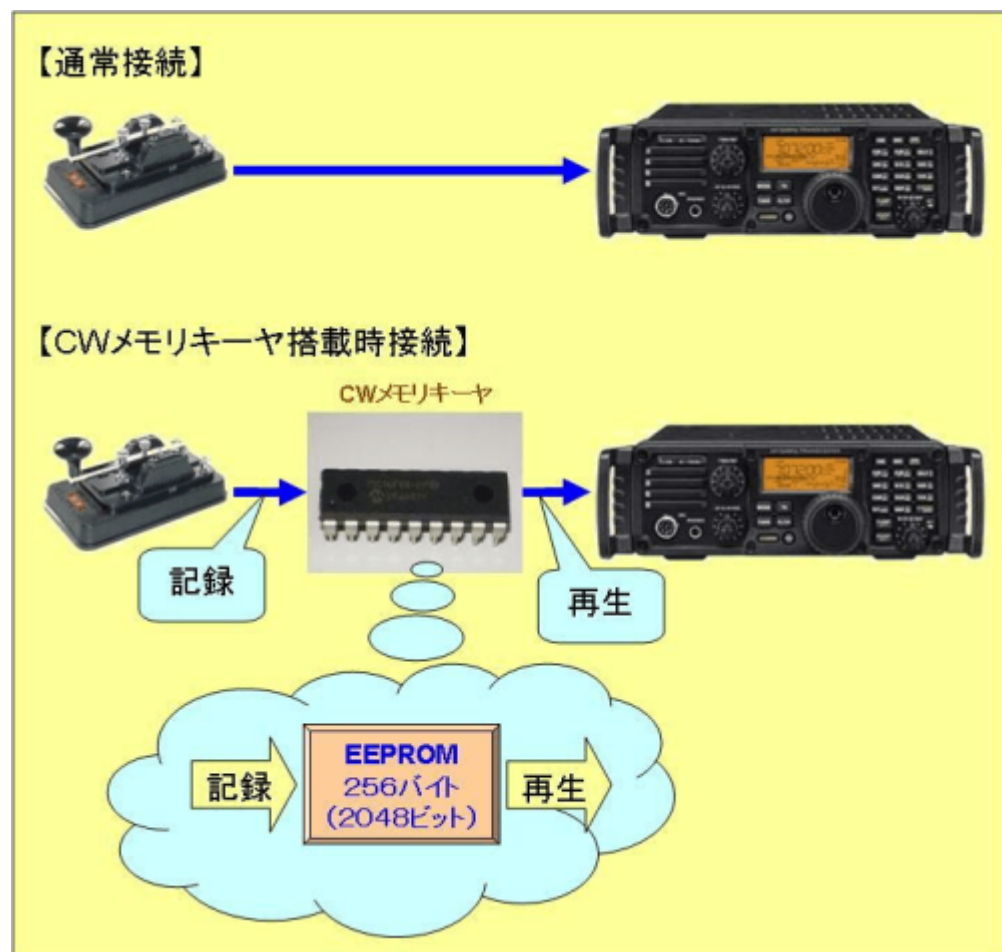
アマチュア無線のコンテストは、一定の時間内にどれだけ沢山の局と交信できるかを競う競技です。CW(モールス信号)の交信では、下文のようなCQを、誰かが呼び出して(CALL)くれるまで、何回も送信することになります。

CQ CQ CQ DE JF3SFB JF3SFB JF3SFB K

数回でコールして頂ければ良いのですが、何十回も送信する場合には、電鍵の操作で腕が疲れてしまいます。そこで、事前に電鍵操作で上文を記憶させ、後はスイッチを押下するだけで自動的に再生送信してくれる、便利な装置(CWメモリーキーヤ)を製作しました。

<仕様>

- 記録時間を切り替え可能とする。(精度高:約40秒、精度低:約60秒)
- EEPROMに記録し、電源をOFFにしても記録内容が消えないようにする。
- モールス音(約1kHz)を圧電スピーカで鳴らせる。
- 単三電池2本で動作可能とする。



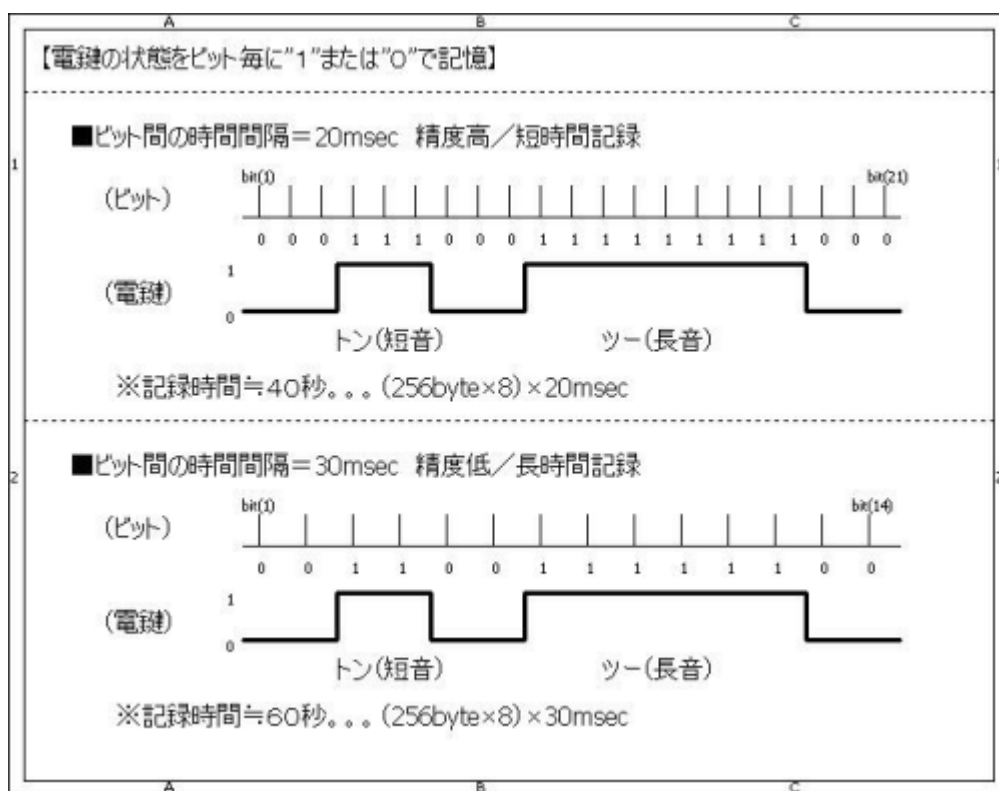
動作原理

<記録>

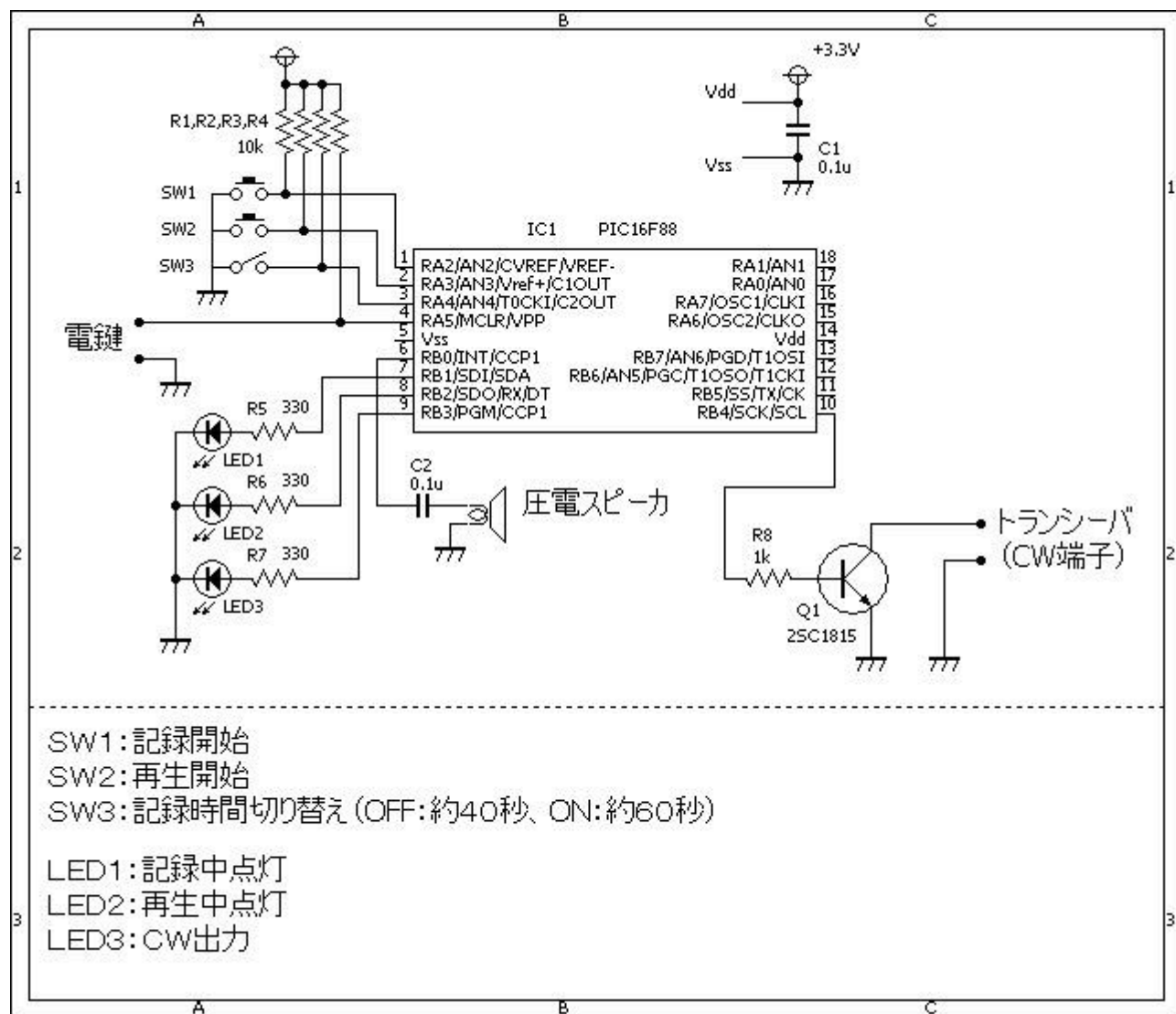
- 電鍵の操作(ON/OFF)を、PIC16F88内蔵のEEPROM(256バイト:2048ビット)に順次記録します。
- 電鍵の操作で、“ON”時には、ビットを“1”にし“OFF”時には、ビットを“0”にします。
- 電鍵の操作で、“ON”時には、ブザー(約1kHz)を鳴らせLEDを点灯させます。
- 記録時の時間間隔は、スイッチ設定で、20msecまたは30msecを切り替えます。
- 従って、記録時間は、約40秒(2048ビット×20msec)または約60秒(2048ビット×30msec)となります。

<再生>

- EEPROMに記録された内容を順次取り出します。
- ビットが、“1”であれば、ブザー(約1kHz)を鳴らせLEDを点灯させ、トランジスタを、“ON”にします。
- トランジスタが、“ON”になると、エミッタ/コレクタ間が導通状態となるため、無線機から見ると、電鍵が“ON”になった状態と同じになります。



回路図



ソースコード

[cw_memory_keyer.c](#)

```
//*****
*
/*
   □□□メモリーキーヤ>
*/
//*****
*

#define SW_REC          PORTA.F2
#define SW_PLAY        PORTA.F3
#define SW_ACCURACY     PORTA.F4

#define LED_REC         PORTB.F1
#define LED_PLAY        PORTB.F2
#define LED_OUTPUT      PORTB.F3
```

```
#define      CW_INPUT      PORTA.F5
#define      CW_OUTPUT     PORTB.F4

#define      DELAY_TIME1   20
#define      DELAY_TIME2   30

//*****
*

void  Pwm_Change_DutyEx(unsigned int duty_ratio)
{
    CCP1L = duty_ratio >> 2;
    CCP1CON.F6 = duty_ratio & 0b00000001;
    CCP1CON.F7 = (duty_ratio & 0b00000010) >> 1;
}

//*****
*

void  Delay_ex()
{
    if (SW_ACCURACY == 1) {
        Delay_ms(DELAY_TIME1);
    } else {
        Delay_ms(DELAY_TIME2);
    }
}

//*****
*

void  recProc()
{
    static unsigned int    cnt1;
    static unsigned short  cnt2, tmp;
    //
    LED_REC = 1;
    for (cnt1 = 0; cnt1 < 256; cnt1++) {
        tmp = 0x00;
        for (cnt2 = 0; cnt2 < 8; cnt2++) {
            if (CW_INPUT == 0) {
                tmp |= 0x01;
                Pwm_Start();
                LED_OUTPUT = 1;
                CW_OUTPUT = 1;
            } else {
                Pwm_Stop();
                LED_OUTPUT = 0;
                CW_OUTPUT = 0;
            }
            if (cnt2 < 7) {
```

```
        tmp = tmp << 1;
        Delay_ex();
    }
}
Eeprom_Write(cnt1, tmp);
Delay_ex();
}
LED_REC = 0;
Pwm_Stop();
LED_OUTPUT = 0;
CW_OUTPUT = 0;
}

//*****
*

void playProc()
{
    static unsigned int cnt1;
    static unsigned short cnt2, tmp;
    //
    LED_PLAY = 1;
    for (cnt1 = 0; cnt1 < 256; cnt1++) {
        tmp = Eeprom_Read(cnt1);
        for (cnt2 = 0; cnt2 < 8; cnt2++) {
            if ((tmp & 0x80) != 0) {
                Pwm_Start();
                LED_OUTPUT = 1;
                CW_OUTPUT = 1;
            } else {
                Pwm_Stop();
                LED_OUTPUT = 0;
                CW_OUTPUT = 0;
            }
            tmp = tmp << 1;
            Delay_ex();
        }
    }
    LED_PLAY = 0;
    Pwm_Stop();
    LED_OUTPUT = 0;
    CW_OUTPUT = 0;
}

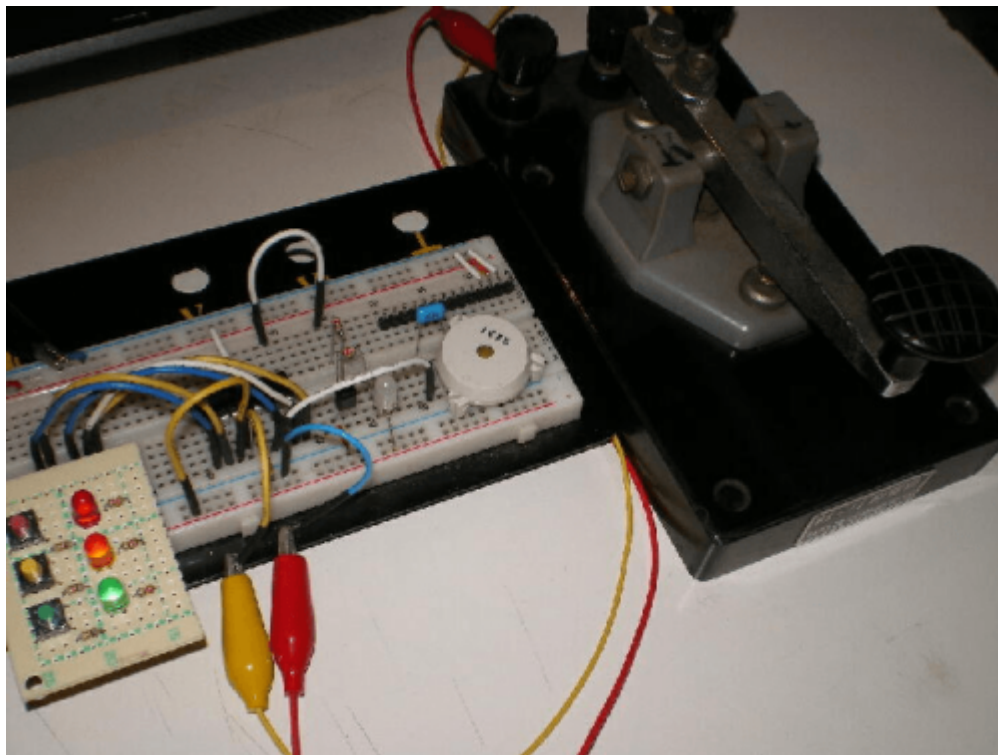
//*****
*

void main()
{
    static short cnt;
    //
```

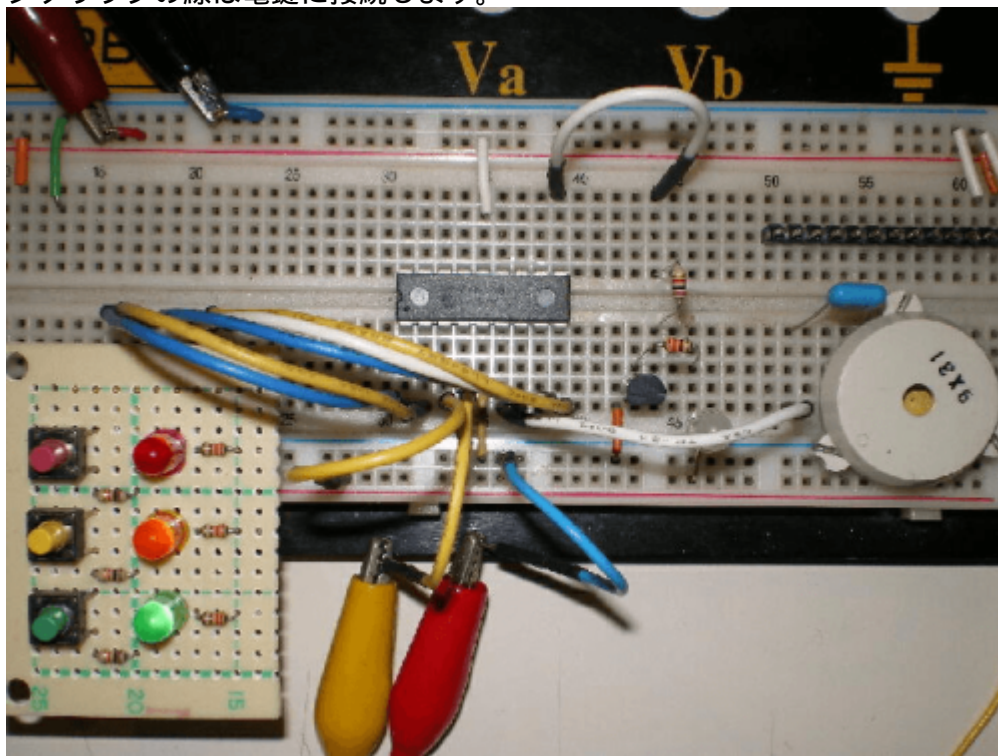
```
//ポート関連の設定
TRISA = 0b11111111;
TRISB = 0b00000000;
OSCCON = 0b01100000; // クロックを4Mhzに設定する。
ANSEL = 0b00000000; // □□□変換は使用しない。
//PWMの設定□1kHz□
Pwm_Init(1000);
Pwm_Change_DutyEx((PR2 * 4) / 2);
Pwm_Stop();
//
for (cnt = 0; cnt < 5; cnt++) {
    LED_REC = 1;
    LED_PLAY = 1;
    LED_OUTPUT = 1;
    Pwm_Start();
    Delay_ms(200);
    Pwm_Stop();
    LED_REC = 0;
    LED_PLAY = 0;
    LED_OUTPUT = 0;
    Delay_ms(200);
}
//
while (1) {
    if (SW_REC == 0) {
        recProc();
    }
    if (SW_PLAY == 0) {
        playProc();
    }
}

//*****
*
```

動作確認



左側から、プッシュスイッチ、LED、PIC16F88、トランジスタ、圧電スピーカです。黄色と赤色のミノムシクリップの線は電鍵に接続します。



著作権表示 copyright notice

このページは稲崎様の閉鎖したHPのコピーで、著作権は稲崎様にあります。[詳細](#) This page is a copy of Mr. Inasaki's closed website, and the copyright is held by him.[Details](#)

From:

<http://www.deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://www.deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:114>

Last update: **2020/05/01 17:41**

