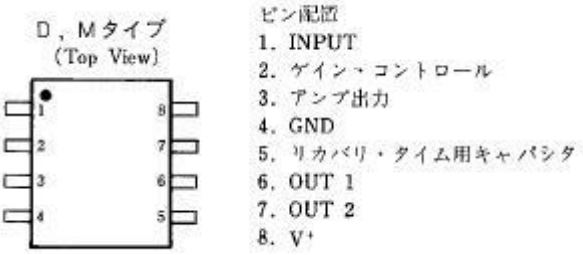
音声スイッチ(VOX)

VOX(Voice Operated X)とは、音声で自動的に送信機を働かせる回路の事を言います。 つまり、手を使わないで無線機器の送受信を切り替える回路です VOXは、無線機器だけでなく、色々な用途に使用されています。(例えば、テープレコーダで音が入ってきたとき だけ録音する等)

VOXとして使用される専用のICとしてNJM2072があります。 今回は、このIC相当(若干機能強化版)の機能を有する「音声スイッチ(VOX)を製作します。

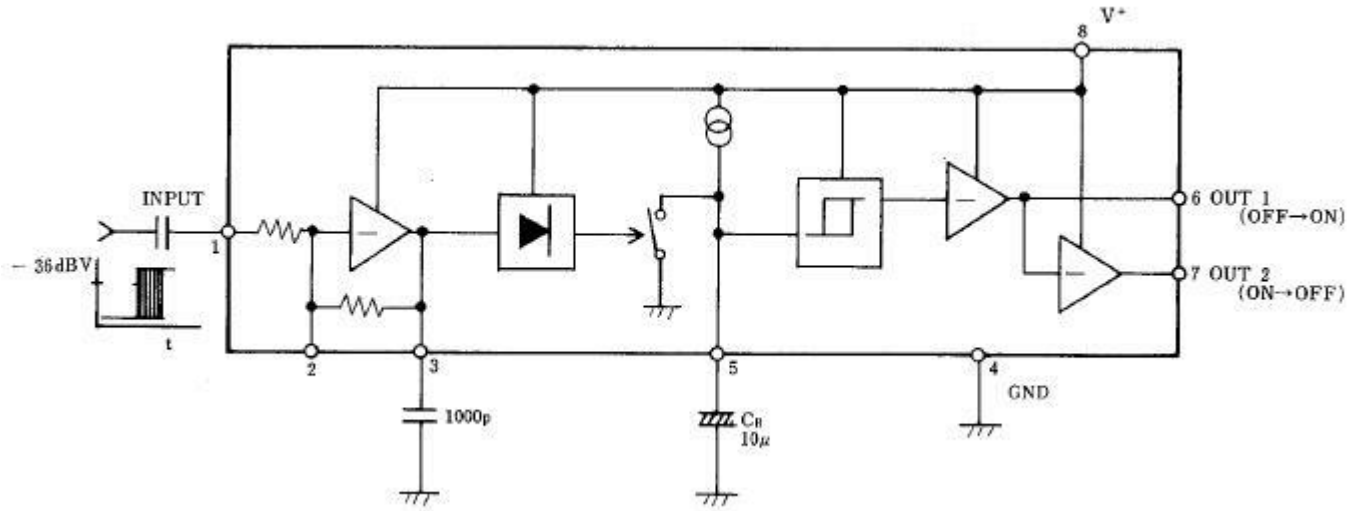
<NJM2072の特長> 音声レベル検出用集積回路です。マイクロカセットの音声検出用、通信機のVOXに最適です。

- 動作電源電圧(0.9~7V)
- 低消費電流(0.55mA)
- 高入力感度(-36dB)

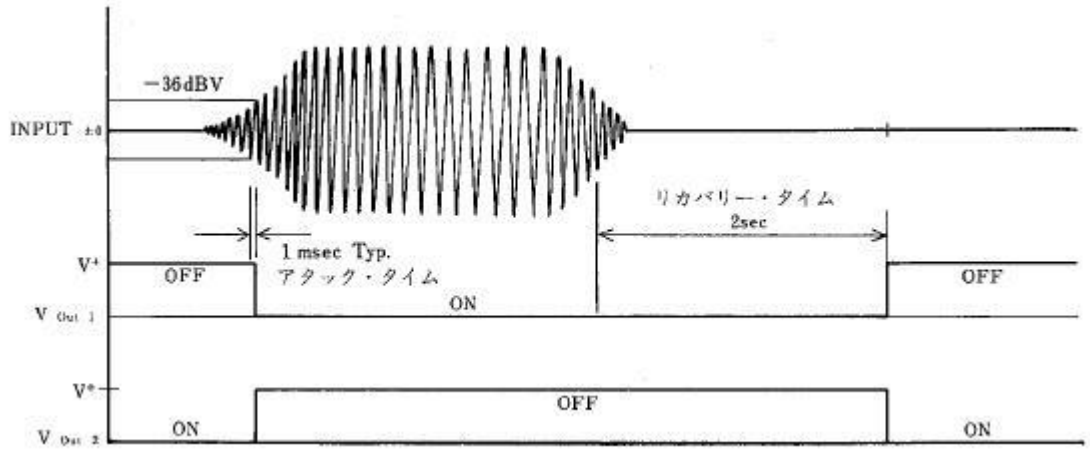


<NJM2072のピン配置>

<NJM2072のブロック図>



<NJM2072のタイムチャート>



動作原理

<仕様>

- 入力の感度切り替え(10段階)
- リカバリータイム切り替え(10段階)。。。0.1秒~1.0秒
- 出力(トランジスタによるオープンコレクタ出力)

<マイク> 信号(音声)の入力には、コンデンサマイクを使用します。 尚、無線機などで使用する際には、高周波の回り込みを防止するために、チョークコイル(数百uH程度)を 使用します。 また、デジタル回路とアナログ回路が共存するので、ノイズ防止のために、コンデンサマイクへの電源供給に 際しては、チョークコイル(1mH~10mH)による平滑回路を使用しました。

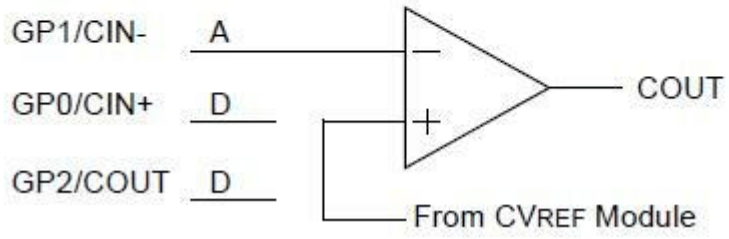
<増幅> コンデンサマイクからの信号を、LM386(低電圧オーディオ・パワーアンプ)を使用し、約200倍の増幅率を得ま す。感度が高すぎる場合には、増幅率を20倍~200倍の間で調整してください。

<整流> 200倍に増幅された信号を、ダイオードを使用し、倍電圧整流します。

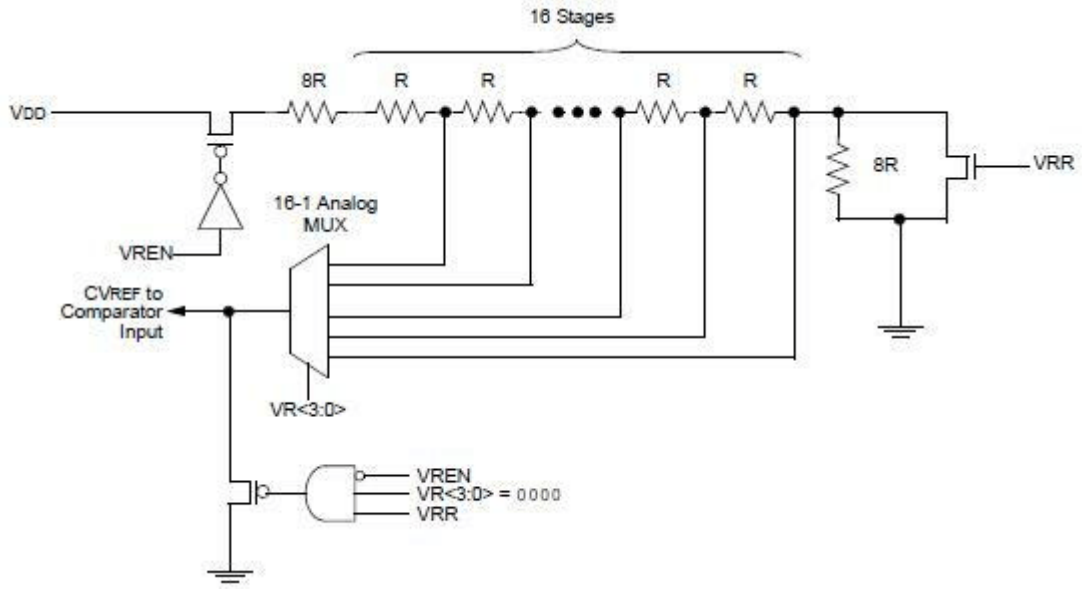
<比較> 整流された信号と、内部の基準電圧を比較し、指定されたレベルに達したかを判断します。 基準電圧との比較は、PIC内蔵の、コンパレータを使用します。

Comparator w/o Output and with Internal Reference

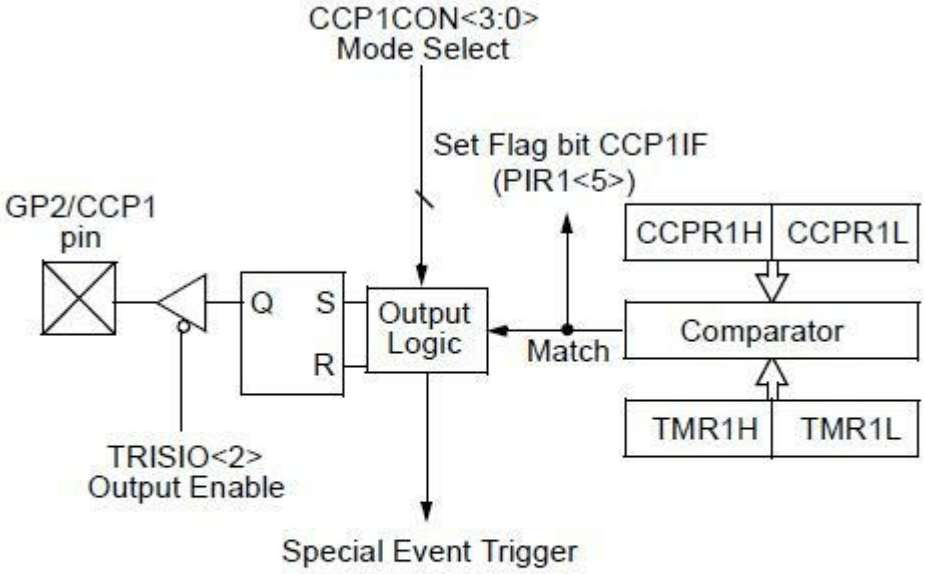
CM<2:0> = 100



<基準電圧> 基準電圧は、PIC内蔵の電圧発生モジュール(COMPARATOR VOLTAGE REFERENCE)を使用します。 16段階の電圧を発生できますが、今回はその内の10段階を、感度調整用として使用します。



<リカバリータイム> 入力信号(音声)が、指定されたレベルに達すると、ある一定時間(リカバリータイム)だけ、出力をONします。 リカバリータイムは、0.1秒~1.0秒迄の10段階(0.1秒単位)で設定可能としました。 0.1秒の時間を得るために□PIC内蔵のCPP(Capture/Compare/PWM)モジュールを、コンペアモードで使用します。



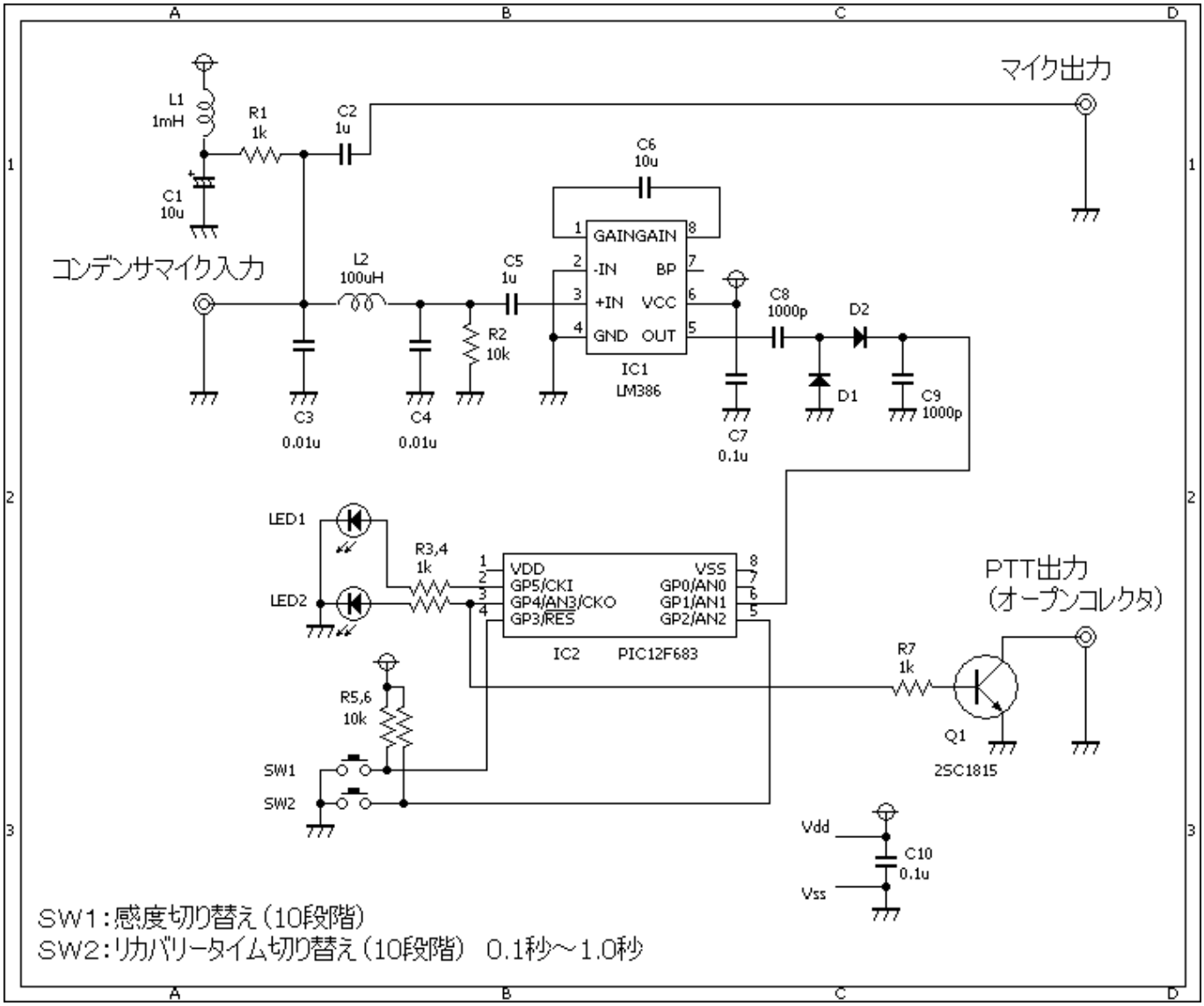
Special Event Trigger will:

- Clear TMR1H and TMR1L registers
- NOT set interrupt flag bit TMR1F (PIR1<0>)
- Set the GO/$\overline{\text{DONE}}$ bit (ADCON0<1>)

<出力> 音声スイッチとしての出力は、LED2およびトランジスタ(オープンコレクタ)をON/OFFします。

回路図

SW1を押下することにより、感度を10段階に切り替えることができます。 (押下する毎に、LED1が点灯し、低感度→高感度→低感度。。。。。に切り替わっていきます) SW2を押下することにより、リカバリータイムを10段階に切り替えることができます。 (押下する毎に、LED1が点灯し、短時間→長時間→短時間。。。。。に切り替わっていきます)



ソースコード

vox_v1.c

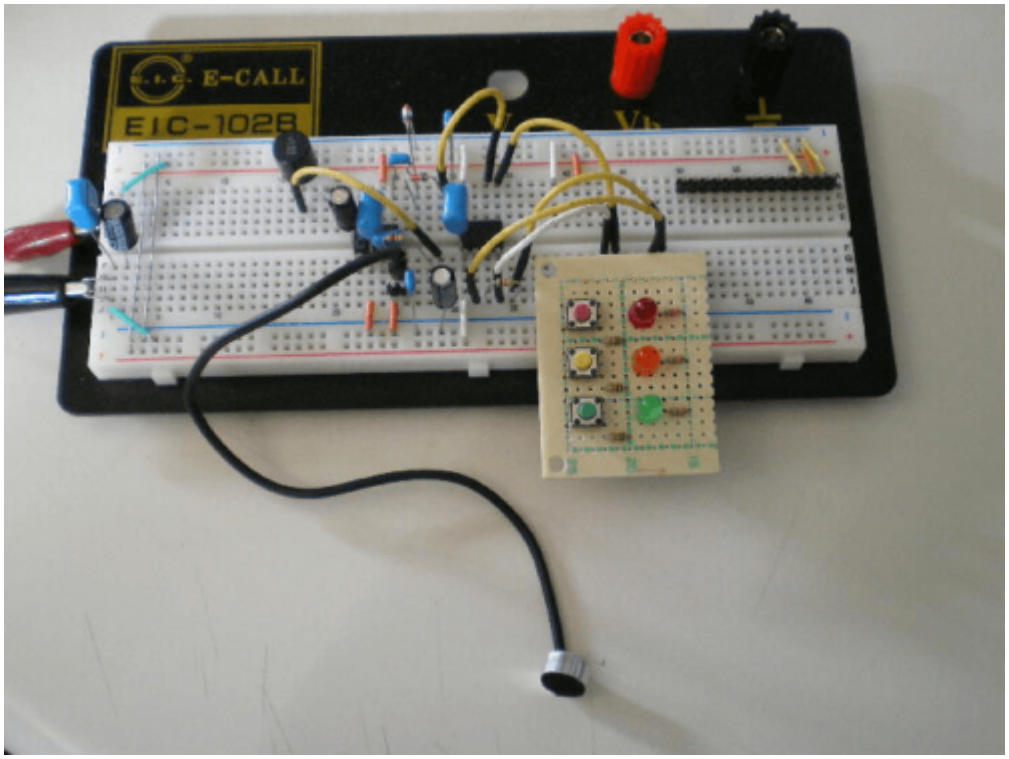
```
//*****************************************************************************
*
/*
    ＜音声スイッチ□□□□□□
*/
//*****************************************************************************
*

#define       LED1    GPIO.F5
#define       LED2    GPIO.F4

#define       SW1        GPIO.F3
#define       SW2        GPIO.F2
```

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

short  on_cnt, threshold;

void   interrupt()
{
    if (PIR1.CMIF == 1) {
        PIR1.CMIF = 0;
        //
        on_cnt = threshold;
    }
    //
    if (PIR1.CCP1IF == 1) {
        PIR1.CCP1IF = 0;
        //
        if (on_cnt > 0) {
            LED2 = 1;
            on_cnt--;
        } else {
            LED2 = 0;
        }
    }
}

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

void main()
{
    static    unsigned    short    Sensitivity, RecoveryTime, 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 = 0b00001110;
    GPIO = 0b00000000;
    // 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;
```

```
    //コンパレータの設定
    CMCON0.CINV = 1;
    PIE1.CMIE = 1;
    PIR1.CMIF = 0;
    //基準電圧の設定
    VRCON.VREN = 1;
    VRCON.VRR = 1;
    VRCON.VR3 = 0;
    VRCON.VR2 = 1;
    VRCON.VR1 = 0;
    VRCON.VR0 = 1;
    //
    LED1 = 0;
    LED2 = 0;
    //
    Sensitivity = 5;
    RecoveryTime = 5;
    threshold = 50;
    //
    INTCON.PEIE = 1;     // これ以降の処理で割り込みを許可する。
    INTCON.GIE = 1;          // これ以降の処理で割り込みを許可する。
    //
    while (1) {
        //感度調整（10段階切替）
        if (SW1 == 0) {
            while (SW1 == 0) {
                Delay_ms(10);
            }
            //
            Sensitivity++;
            Sensitivity = (Sensitivity < 11) ? Sensitivity: 1;
            for (cnt = 0; cnt < Sensitivity; cnt++) {
                LED1 = 1;
                Delay_ms(100);
                LED1 = 0;
                Delay_ms(100);
            }
            //
            VRCON.VR3 = Sensitivity.F3;
            VRCON.VR2 = Sensitivity.F2;
            VRCON.VR1 = Sensitivity.F1;
            VRCON.VR0 = Sensitivity.F0;
        }
        //リカバリータイム調整（0.1秒～1.0秒迄の10段階切替）
        if (SW2 == 0) {
            while (SW2 == 0) {
                Delay_ms(10);
            }
            //
            RecoveryTime++;
```

```
            RecoveryTime = (RecoveryTime < 11) ? RecoveryTime: 1;
            for (cnt = 0; cnt < RecoveryTime; cnt++) {
                LED1 = 1;
                Delay_ms(100);
                LED1 = 0;
                Delay_ms(100);
            }
            //
            threshold = RecoveryTime * 10;
        }
    }
}

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

動作確認



左側から□LM386□ダイオード□PIC12F683□SW□LEDです。