

アナログデータ記録再生ユニット(PIC16F877)

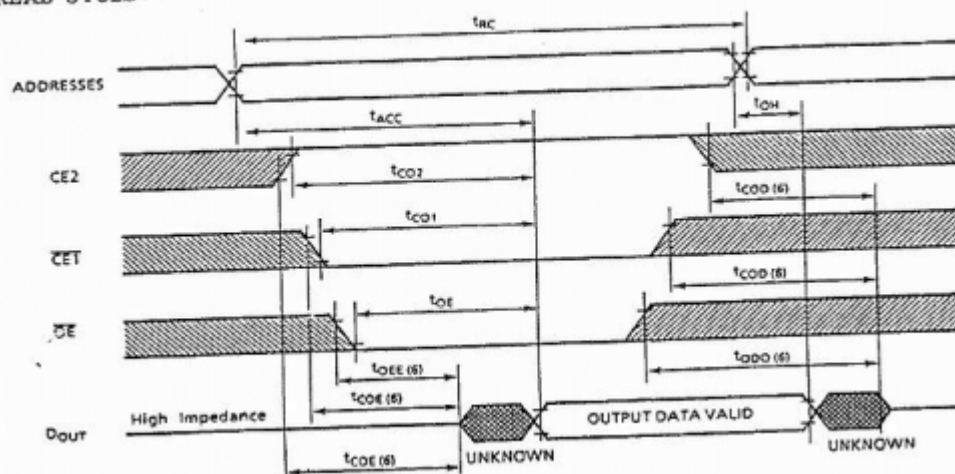
概要

アナログデータを記録し、再生するユニットを製作しました。記録する容量は、約8kバイト(正確には、8192バイト)で、精度は8ビットです。電池駆動にして、外で計測したデータを持ち帰って分析するのに都合が良いと思います。

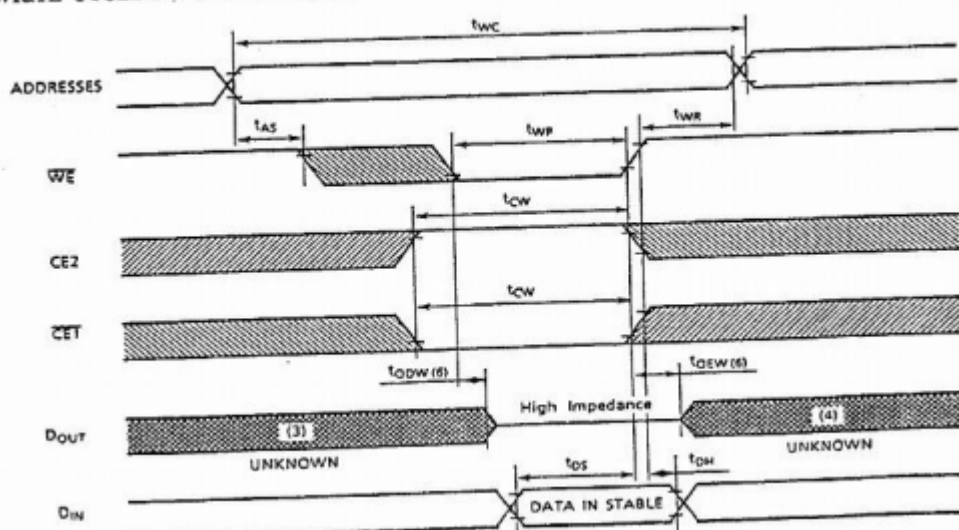
動作原理

- 記録する媒体にはTC5588P(S-RAM)を使用しました。
読み込みと書き込みのタイミングは下図のようになります。
今回は、CE(ChipEnable)端子はPICで制御せずに固定配線としました。

READ CYCLE (2)



WRITE CYCLE 1 (5) (WE Controlled Write)



- TC5588Pのアドレスは13本必要となりPICだけではI/O端子が足りないのでTC4040(バイナリカウンタ)を使用しました。
- D/A変換は馴染みの「R-2R型抵抗ラダー」を使用しました。
- プログラム上ではREC-SWによる、記録処理とPLAY-SWによる再生処理を行います。尚、S-RAMの制御は汎用性を考えサブルーチン化しました。

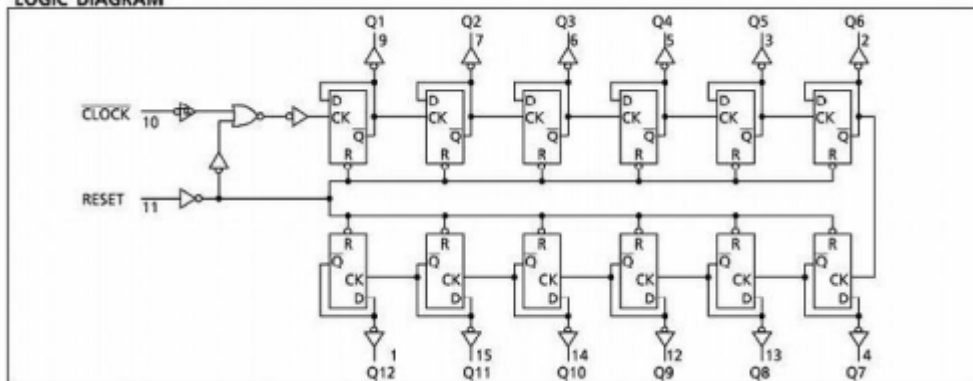
TC5588P(8K×8バイトのS-RAM)です。鈴商で確か150円~200円で販売していたと思います。



TC4040(12ステージのバイナリカウンタ)です。



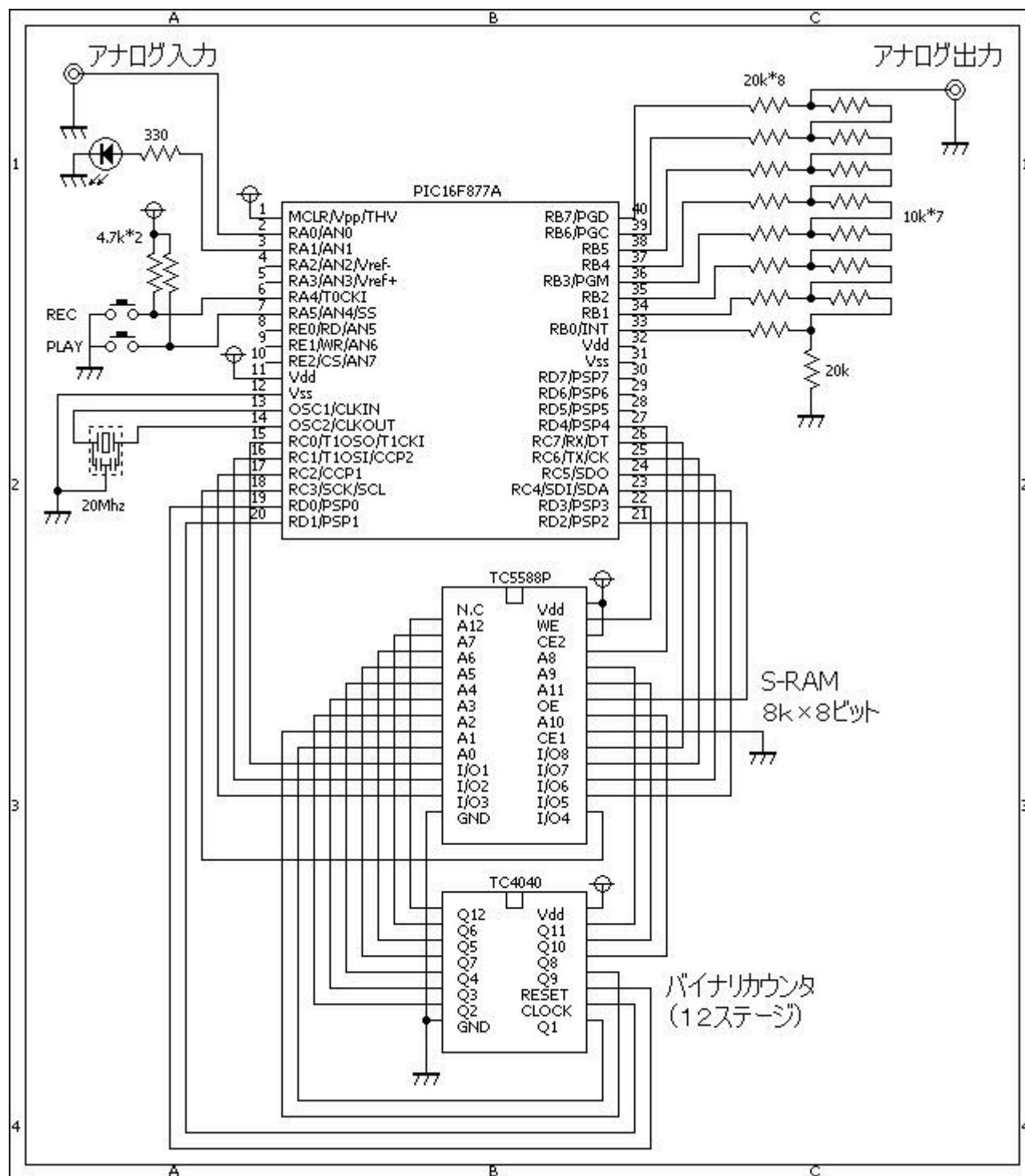
LOGIC DIAGRAM



TC5588P□TC4040□PIC16F877Aの大きさを比較してみました。



回路図



ソースコード

VoiceRec2.c

```
//*****  
*  
  
#define LED PORTA.F1  
  
#define BANK PORTD.F4
```

```
#define      CLOCK      PORTD.F1
#define      RESET      PORTD.F0
#define      WE          PORTD.F3
#define      OE          PORTD.F2

#define      REC          PORTA.F4
#define      PLAY      PORTA.F5

//*****
*

void  sram_reset()
{
    BANK = 0;
    RESET = 1;
    RESET = 0;
    CLOCK = 1;
}

void  sram_increment()
{
    CLOCK = 0;
    CLOCK = 1;
}

void  sram_bank0()
{
    BANK = 0;
}

void  sram_bank1()
{
    BANK = 1;
}

void  sram_setData(unsigned char data)
{
    WE = 0;
    TRISC = 0b00000000;
    PORTC = data;
    WE = 1;
    sram_increment();
}

unsigned char  sram_getData()
{
    TRISC = 0b11111111;
    OE = 0;
    OE = 1;
    sram_increment();
    return(PORTC);
}
```

```
}

char    sram_check()
{
    unsigned    int        cnt;
    //
    sram_bank0();
    for (cnt = 0; cnt < 4096; cnt++) {
        sram_setData(0xA5);
    }
    sram_bank1();
    for (cnt = 0; cnt < 4096; cnt++) {
        sram_setData(0xA5);
    }
    //
    sram_bank0();
    for (cnt = 0; cnt < 4096; cnt++) {
        if (sram_getData() != 0xA5)
            return(-1);
    }
    sram_bank1();
    for (cnt = 0; cnt < 4096; cnt++) {
        if (sram_getData() != 0xA5)
            return(-1);
    }
    //
    return(0);
}

//*****
*

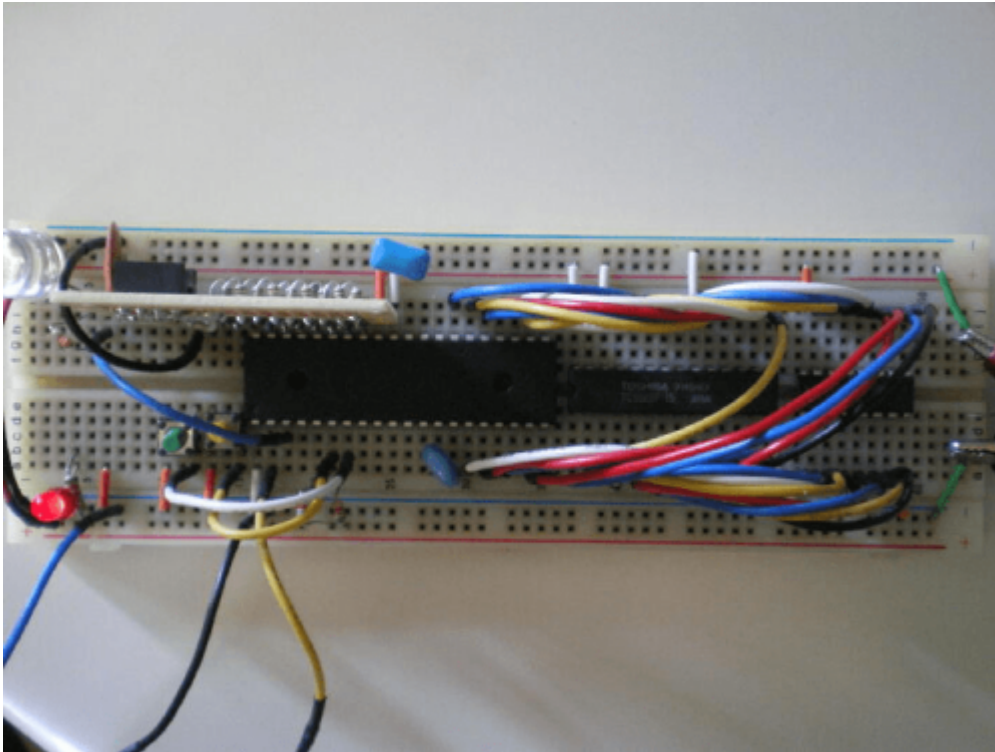
void    main()
{
    unsigned    int        cnt, ad0;
    //
    TRISA = 0b00110001;
    TRISB = 0b00000000;
    TRISC = 0b11111111;
    TRISD = 0b00000000;
    TRISE = 0b00000000;
    ADCON1.PCFG0 = 0;
    ADCON1.PCFG1 = 1;
    ADCON1.PCFG2 = 1;
    ADCON1.PCFG3 = 1;
    CMCON.CM0 = 1;
    CMCON.CM1 = 1;
    CMCON.CM2 = 1;
    WE = 1;
    OE = 1;
```

```
LED = 0;
//
sram_reset();
if (sram_check() == 0) {
    LED = 1;
    Delay_ms(500);
    LED = 0;
}
//
while (1) {
    if (REC == 0) {
        sram_bank0();
        for (cnt = 0; cnt < 4096; cnt++) {
            ad0 = Adc_Read(0);
            sram_setData(ad0 >> 2);
        }
        sram_bank1();
        for (cnt = 0; cnt < 4096; cnt++) {
            ad0 = Adc_Read(0);
            sram_setData(ad0 >> 2);
        }
        //
        LED = 1;
        Delay_ms(500);
        LED = 0;
    }
    //
    if (PLAY == 0) {
        sram_bank0();
        for (cnt = 0; cnt < 4096; cnt++) {
            PORTB = sram_getData();
        }
        sram_bank1();
        for (cnt = 0; cnt < 4096; cnt++) {
            PORTB = sram_getData();
        }
    }
}

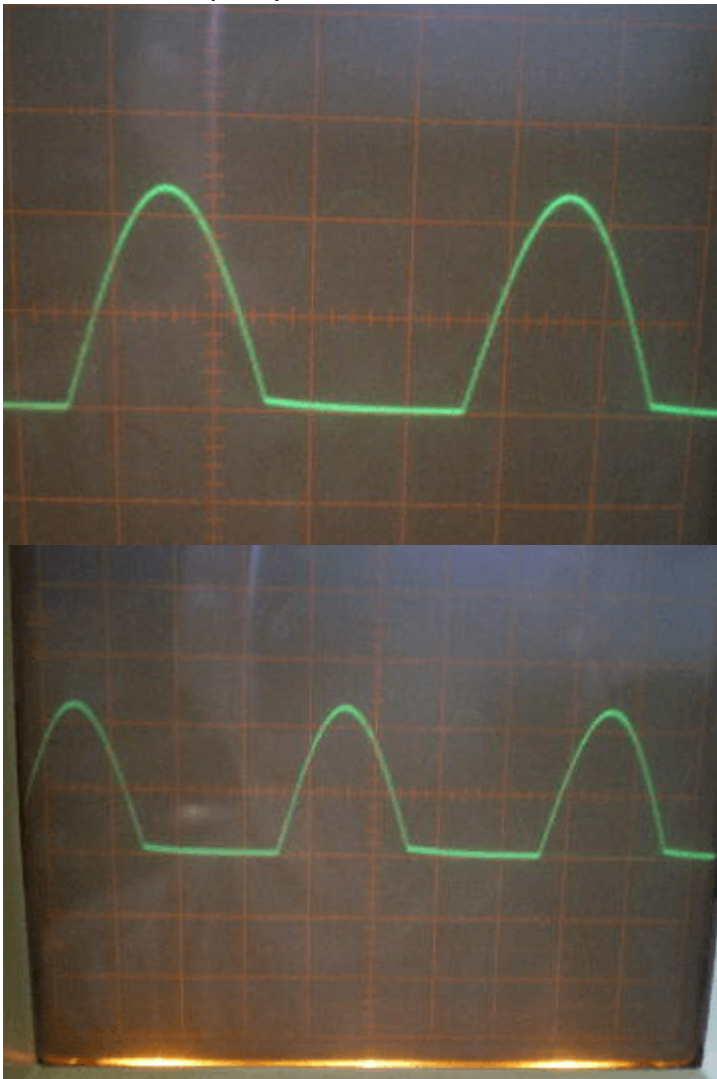
//*****
*
```

動作確認

いつものブレッドボードで確認しました。結構配線が多いので出来るだけ交差しないような配置にしました。黄色のプッシュスイッチが、記録(REC)用です。緑色のプッシュスイッチが、再生(PLAY)用です。



100Hzの正弦波(半波)を入力し、記録(REC)したものを再生(PLAY)してみました。



From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:otherpic:170&rev=1588238479>

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

