

アナログデータロガーV2(携帯型)

概要

以前にもパソコン直結型のアナログデータロガーを製作しましたが、今回はデータ収集時には、特にパソコンを必要としない携帯型のアナログデータロガーを製作しました。

<仕様>

- A/D変換の精度は10ビット、最大電圧は約2.5V、最小分解能は約2.4mV
- 最大16384のデータを記録
- データ記録周期は、1秒(記録時間:約4.5時間)と1分(記録時間:約11日)の切り替えが可能
- データ記録時には、パソコン不要の携帯型
- データ再生時には、パソコンとRS232Cで接続し、9600bpsの速度で転送可能

動作原理

PIC16F88の内蔵EEPROMは、256バイトしかありませんので、外付けのEEPROMを使用します。I2CインターフェースのEEPROMはいろいろありますが、今回は、コストパフォーマンスの高い、256kバイトのEEPROMを使用しました。現時点の価格は、秋月電子で160円です。

<EEPROM(24LC256/24C256)の特長>

- 256Kビット(32kx8)EEPROM
- 2線式I2Cインターフェース
- 動作電圧:2.7~5.5V
- 8ピンDIP

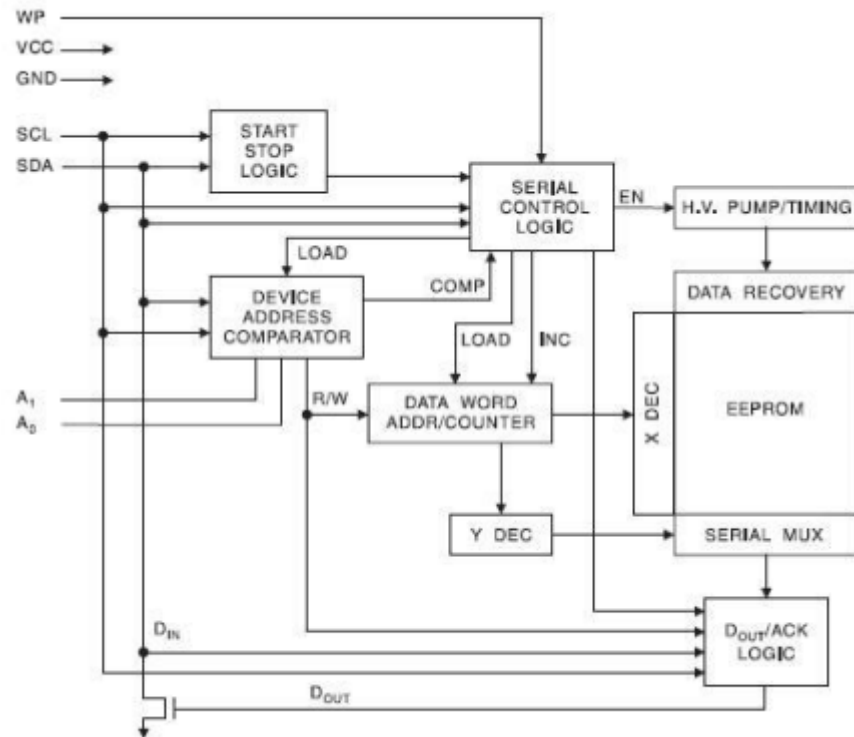
<記録処理の流れ>

1. PIC16F88のA/D変換モジュールでアナログデータを取り込む。
2. 取り込んだデータをEEPROM(24C256)に記録する。
3. 取り込んだデータをLCDに表示する。
4. 2.3.を16384回繰り返す。

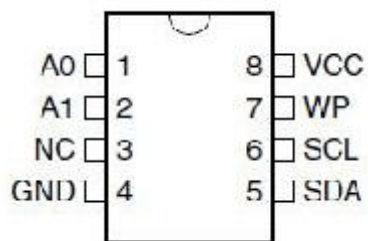
<再生処理の流れ>

1. EEPROM(24C256)に記録したデータを読み出す。
2. 読み出したデータをLCDに表示する。
3. 読み出したデータをRS232C経由でパソコンに転送(9600bps)する。
4. 2.3.を16384回繰り返す。

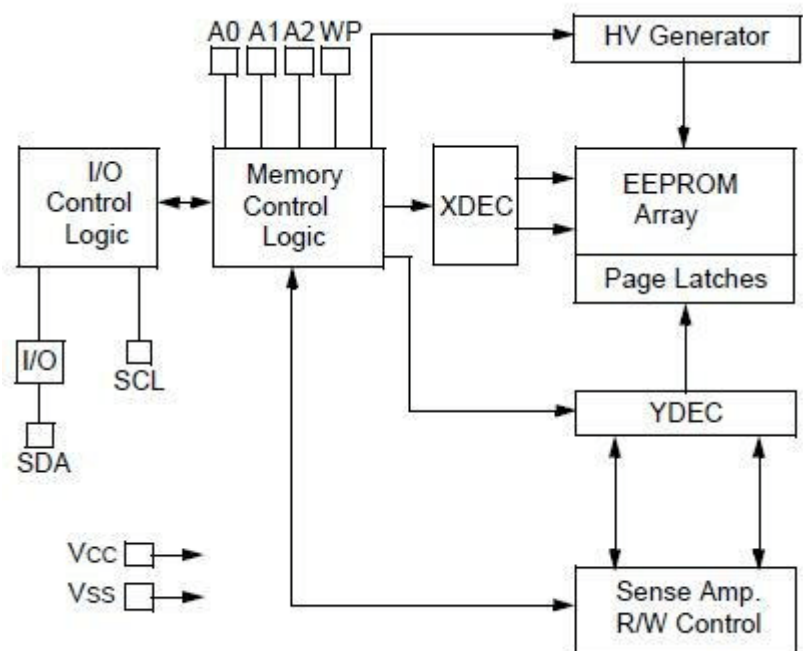
EEPROM(ATMEL社製/24C256)のブロックダイアグラムとピンアサインです。アドレス指定が、A0,A1な



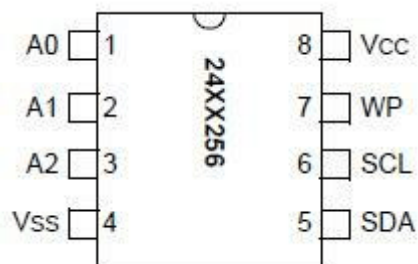
ので4個接続可能です。



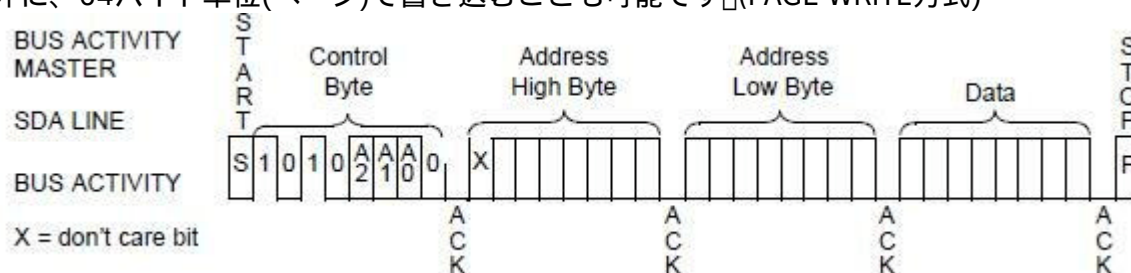
EEPROM(Microchip社製/24LC256)のブロックダイアグラムとピンアサインです。 アドレス指定



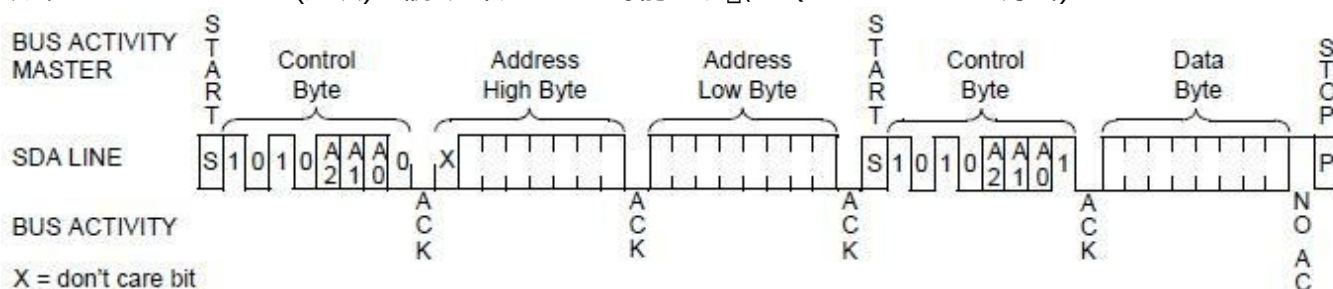
が、A0,A1,A2なので8個接続可能です。



EEPROM(24LC256・24C256)へのアドレス指定によるバイトデータ書き込みフォーマットです。これ以外に、64バイト単位(ページ)で書き込むことも可能です(PAGE WRITE方式)

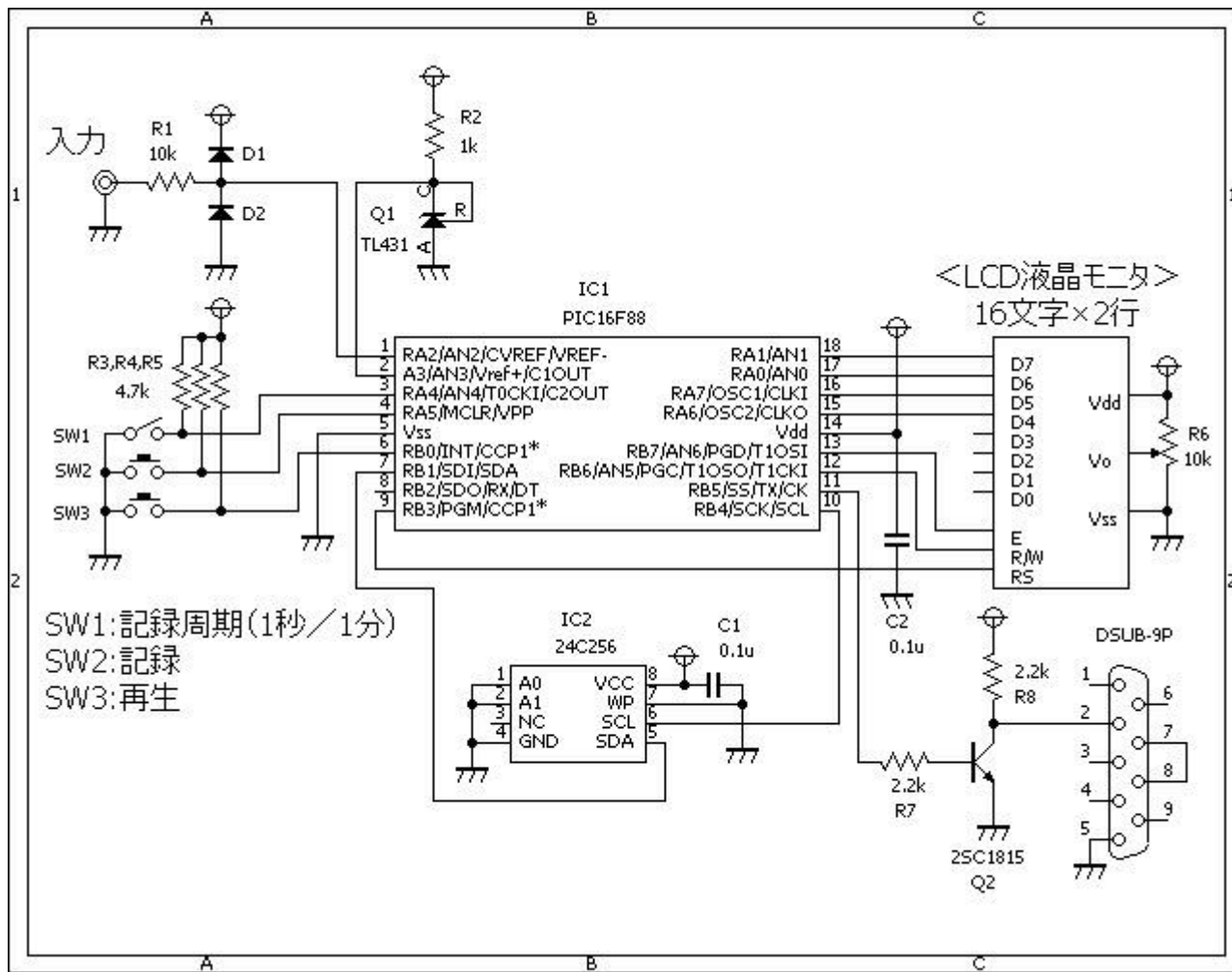


EEPROM(24LC256・24C256)からのアドレス指定によるバイトデータ読み出しフォーマットです。これ以外にシーケンシャル(逐次)に読み出すことも可能です(SEQUENTIAL READ方式)



回路図

Microchip社製/24LC256は、ATMEL社製/24C256のNCピン(3)が、A2となるのでGNDに接続します。



ソースコード

DataLoggerV2old.c

```

//*****
*
/*
□DataLoggerV2□
■ アナログデータの記録機能および再生機能を提供する。
  記録媒体には、アトメル社製の□□□□□□□□を使用する。
  □□□□□□ビット□□□□□□□□□□□□□□
  : 2線式□□□□インターフェース
■ 記録チャンネル数は、1CHとする。
■ データの精度は、10ビットとする。
■ 記録容量は、
  16384件(1件あたり2バイト使用□□□□÷□□)
■ 記録時間は、
  記録周期が1秒の場合は、約4.5時間の記録が可能
  記録周期が1分の場合は、約1日間の記録が可能
■ パソコンとの通信は、RS232Cで9600bpsの速度で行う。
*/
//*****
*

```

```
#define      SW_CYCLE      PORTA.F4

#define      SW_REC        PORTA.F5
#define      SW_PLAY       PORTB.F0

//*****
*

static short  TimerCnt;

void  interrupt()
{
    if (PIR1.CCP1IF == 1) {          //0.1秒周期の割り込みが発生する。
        PIR1.CCP1IF = 0;
        //
        if (TimerCnt > 0)
            TimerCnt--;
    }
}

//*****
*

void  Delay_1sec()
{
    TimerCnt = 10;
    while(TimerCnt != 0)
        ;
}

//*****
*

void  Delay_1min()
{
    TimerCnt = 600;
    while(TimerCnt != 0)
        ;
}

//*****
*

void  Uart_Write_String(char *buf)
{
    static  int    len, i;
    len = strlen(buf);
    for (i = 0; i < len; i++) {
        Uart_Write(buf[i]);
    }
}
```

```

}

//*****
*

void    EEPROM_24C256_Init()
{
    Soft_I2C_Config(&PORTB, 1, 4);
}

//*****
*

unsigned short    EEPROM_24C256_Read(unsigned int    addr)
{
    unsigned short    data;
    //
    Soft_I2C_Start();
    Soft_I2C_Write(0xA0);
    Soft_I2C_Write((addr >> 8) & 0xFF);
    Soft_I2C_Write(addr & 0xFF);
    Soft_I2C_Start();
    Soft_I2C_Write(0xA1);
    data = Soft_I2C_Read(0);    //not acknowledge
    Soft_I2C_Stop();
    //
    return (data);
}

//*****
*

void    EEPROM_24C256_Write(unsigned int addr, unsigned short data)
{
    Soft_I2C_Start();
    Soft_I2C_Write(0xA0);
    Soft_I2C_Write((addr >> 8) & 0xFF);
    Soft_I2C_Write(addr & 0xFF);
    Soft_I2C_Write(data);
    Soft_I2C_Stop();
    Delay_ms(5);
}

//*****
*

void    Data_Rec()    //データを記録する。
{
    static unsigned int    ad, cnt;
    static unsigned char    buf[10];
    //

```

```

    for (cnt = 0; cnt < 32768; cnt++) {
        // アナログデータの取り込み
        ad = Adc_Read(2);
        // アナログデータをEEPROMに記録する。
        EEPROM_24C256_Write(cnt, (ad >> 8) & 0xFF);
        cnt++;
        EEPROM_24C256_Write(cnt, ad & 0xFF);
        // カウンタ値をLCDに表示する。
        WordToStr((cnt / 2) + 1, buf);
        Lcd_Custom_Out(2, 9, buf);
        // アナログデータをLCDに表示する。
        WordToStr((double)ad * 2.4365234375, buf);    // 2.4365234375 □
2.495V ÷ 1024
        Lcd_Custom_Out(2, 1, buf);
        Lcd_Custom_Out(2, 6, "mV");
        // 記録周期を確認する
        if (SW_CYCLE == 1) {
            Delay_1sec();    // 1秒周期
        } else {
            Delay_1min();    // 1分周期
        }
        //
        if (SW_REC == 0) {    // 記録を中断するかを判断する。
            return;
        }
    }
}

// *****
*

void Data_Play()    // データを再生する。
{
    static unsigned int ad, cnt;
    static unsigned char buf[10];
    //
    for (cnt = 0; cnt < 32768; cnt++) {
        // EEPROMからアナログデータを取り出す。
        ad = EEPROM_24C256_Read(cnt);
        ad = (ad << 8) & 0xFF00;
        cnt++;
        ad = ad | EEPROM_24C256_Read(cnt);
        // カウンタ値をLCDに表示する。
        WordToStr((cnt / 2) + 1, buf);
        Lcd_Custom_Out(2, 9, buf);
        // アナログデータをLCDに表示する。
        WordToStr((double)ad * 2.4365234375, buf);
        Lcd_Custom_Out(2, 1, buf);
        Lcd_Custom_Out(2, 6, "mV");
        // アナログデータをパソコンへ転送する。
        Usart_Write_String(buf);
    }
}

```

```

    Usart_Write_String("mV\r\n");
    //
    if (SW_PLAY == 0) {    //再生を中断するかを判断する。
        return;
    }
}

//*****
*

void main()
{
    CMCON = 0b00000111;    //コンパレータは使用しない
    ANSEL = 0b00000100;    //A/DコンバータはAD2を使用する。
    ADCON1.VCFG1 = 1;
    ADCON1.VCFG0 = 0;
    OSCCON = 0b01110000;    //クロックは内臓8MHzを使用する。
    TRISA = 0b00111100;    //PORTAを設定する。
    TRISB = 0b00000001;    //PORTBを設定する。
    OPTION_REG.NOT_RBPU = 0;
    //LCDを初期化する。
    Lcd_Custom_Config(&PORTA,1,0,7,6,&PORTB,3,6,7);
    Lcd_Custom_Cmd(LCD_CURSOR_OFF);
    Lcd_Custom_Cmd(LCD_CLEAR);
    //    Lcd_Custom_Out(1, 1, "DataLoggerV2");
    //    Delay_ms(1000);
    //    Lcd_Custom_Cmd(LCD_CLEAR);
    //RS232Cを初期化する。
    Usart_Init(9600);
    //    Usart_Write_String("\r\nDataLoggerV2\r\n");
    //EEPROMを初期化する。
    EEPROM_24C256_Init();
    // CCPを設定する。
    PIE1.CCP1IE = 1;
    PIR1.CCP1IF = 0;
    CCP1CON.CCP1M3 = 1;    //trigger special event mode
    CCP1CON.CCP1M2 = 0;
    CCP1CON.CCP1M1 = 1;
    CCP1CON.CCP1M0 = 1;
    CCPR1L = 0xA8;    // 0.1sec...10hz...クロックが8MHzの時
    CCPR1H = 0x61;    // 0.1sec...(1÷8000000)*4*8*25000
    // TIMER1を設定する。
    PIE1.TMR1IE = 0;
    PIR1.TMR1IF = 0;
    TMR1L = 0;
    TMR1H = 0;
    T1CON.T1CKPS0 = 1;
    T1CON.T1CKPS1 = 1;
    T1CON.TMR1ON = 1;
    //変数を初期化する。

```

```

TimerCnt = 0;
// 割り込み(全体)の設定
INTCON.PEIE = 1;
INTCON.GIE = 1;
//
while (1) {
    // 記録スイッチが押されていれば、アナログデータの記録を行う。
    if (SW_REC == 0) {
        while (SW_REC == 0)
            Delay_ms(100);
        Lcd_Custom_Out(1, 1, "REC ");
        Data_Rec();
        Lcd_Custom_Cmd(LCD_CLEAR);
    }
    // 再生スイッチが押されていれば、アナログデータの再生を行う。
    if (SW_PLAY == 0) {
        while (SW_PLAY == 0)
            Delay_ms(100);
        Lcd_Custom_Out(1, 1, "PLAY");
        Data_Play();
        Lcd_Custom_Cmd(LCD_CLEAR);
    }
    Delay_ms(500);
}

// *****
*
```

記録スイッチ(SW2)および再生スイッチ(SW3)の取り込み部分を見直しました。このプログラムは、試用版の制限を越えるため正式版でなければコンパイルできません。

DataLoggerV2.c

```

// *****
*
/*
DataLoggerV2
■ アナログデータの記録機能および再生機能を提供する。
  記録媒体には、アトメル社製の□□□□□□□□を使用する。
□□□□□□□□ ビット□□□□□□□□□□□□□□
  : 2線式□□□□インターフェース
■ 記録チャンネル数は、1CHとする。
■ データの精度は、10ビットとする。
■ 記録容量は、
  16384件(1件あたり2バイト使用□□□□÷□□
■ 記録時間は、
  記録周期が1秒の場合は、約4.5時間の記録が可能
  記録周期が1分の場合は、約1日間の記録が可能
■ パソコンとの通信は、RS232Cで9600bpsの速度で行う。
*/
```

```
//*****
*

#define      SW_CYCLE      PORTA.F4

#define      SW_REC        PORTA.F5
#define      SW_PLAY       PORTB.F0

#define      ON            1
#define      OFF           0

//*****
*

static short  TimerCnt, swRec, swPlay, swRecCnt, swPlayCnt;

void  interrupt()
{
    if (PIR1.CCP1IF == 1) {          //0.1秒周期の割り込みが発生する。
        PIR1.CCP1IF = 0;
        //
        if (TimerCnt > 0)
            TimerCnt--;
        if (SW_REC == 0) {
            swRecCnt++;
        } else {
            if (swRecCnt > 3)
                swRec = ON;
            swRecCnt = 0;
        }
        if (SW_PLAY == 0) {
            swPlayCnt++;
        } else {
            if (swPlayCnt > 3)
                swPlay = ON;
            swPlayCnt = 0;
        }
    }
}

//*****
*

void  Delay_1sec()
{
    TimerCnt = 10;
    while(TimerCnt != 0) {
        if (swRec == ON)
            return;
    }
}
```

```
//*****
*

void Delay_1min()
{
    TimerCnt = 600;
    while(TimerCnt != 0) {
        if (swRec == ON)
            return;
    }
}

//*****
*

void Usart_Write_String(char *buf)
{
    static int len, i;
    len = strlen(buf);
    for (i = 0; i < len; i++) {
        Usart_Write(buf[i]);
    }
}

//*****
*

void EEPROM_24C256_Init()
{
    Soft_I2C_Config(&PORTB, 1, 4);
}

//*****
*

unsigned short EEPROM_24C256_Read(unsigned int addr)
{
    unsigned short data;
    //
    Soft_I2C_Start();
    Soft_I2C_Write(0xA0);
    Soft_I2C_Write((addr >> 8) & 0xFF);
    Soft_I2C_Write(addr & 0xFF);
    Soft_I2C_Start();
    Soft_I2C_Write(0xA1);
    data = Soft_I2C_Read(0); //not acknowledge
    Soft_I2C_Stop();
    //
    return (data);
}
```

```

//*****
*

void    EEPROM_24C256_Write(unsigned int addr, unsigned short data)
{
    Soft_I2C_Start();
    Soft_I2C_Write(0xA0);
    Soft_I2C_Write((addr >> 8) & 0xFF);
    Soft_I2C_Write(addr & 0xFF);
    Soft_I2C_Write(data);
    Soft_I2C_Stop();
    Delay_ms(5);
}

//*****
*

void    Data_Rec()          //データを記録する。
{
    static    unsigned    int        ad, cnt;
    static    unsigned    char    buf[10];
    //
    for (cnt = 0; cnt < 32768; cnt++) {
        //アナログデータの取り込み
        ad = Adc_Read(2);
        //アナログデータをEEPROMに記録する。
        EEPROM_24C256_Write(cnt, (ad >> 8) & 0xFF);
        cnt++;
        EEPROM_24C256_Write(cnt, ad & 0xFF);
        //カウンタ値をLCDに表示する。
        WordToStr((cnt / 2) + 1, buf);
        Lcd_Custom_Out(2, 12, buf);
        //アナログデータをLCDに表示する。
        WordToStr((double)ad * 2.4365234375, buf);    //2.4365234375 □
        2.495V ÷ 1024
        Lcd_Custom_Out(2, 1, buf);
        Lcd_Custom_Out(2, 6, "mV");
        //記録周期を確認する
        if (SW_CYCLE == 1) {
            Delay_1sec();    //1秒周期
        } else {
            Delay_1min();    //1分周期
        }
        //
        if (swRec == ON) {    //記録を中断するかを判断する。
            swRec = OFF;
            return;
        }
    }
}

```

```

//*****
*

void Data_Play()          //データを再生する。
{
    static unsigned int    ad, cnt;
    static unsigned char   buf[10];
    //
    for (cnt = 0; cnt < 32768; cnt++) {
        //EEPROMからアナログデータを取り出す。
        ad = EEPROM_24C256_Read(cnt);
        ad = (ad << 8) & 0xFF00;
        cnt++;
        ad = ad | EEPROM_24C256_Read(cnt);
        //カウンタ値をLCDに表示する。
        WordToStr((cnt / 2) + 1, buf);
        Lcd_Custom_Out(2, 12, buf);
        //アナログデータをLCDに表示する。
        WordToStr((double)ad * 2.4365234375, buf);
        Lcd_Custom_Out(2, 1, buf);
        Lcd_Custom_Out(2, 6, "mV");
        //アナログデータをパソコンへ転送する。
        Uart_Write_String(buf);
        Uart_Write_String("mV\r\n");
        //
        if (swPlay == ON) {      //再生を中断するかを判断する。
            swPlay = OFF;
            return;
        }
    }
}

//*****
*

void main()
{
    CMCON  = 0b00000111;          //コンパレータは使用しない
    ANSEL  = 0b00000100;          //A/DコンバータはAD2を使用する。
    ADCON1.VCFG1 = 1;
    ADCON1.VCFG0 = 0;
    OSCCON = 0b01110000;          //クロックは内臓8MHzを使用する。
    TRISA  = 0b00111100;          //PORTAを設定する。
    TRISB  = 0b00000001;          //PORTBを設定する。
    OPTION_REG.NOT_RBPU = 0;
    //LCDを初期化する。
    Lcd_Custom_Config(&PORTA, 1, 0, 7, 6, &PORTB, 3, 6, 7);
    Lcd_Custom_Cmd(LCD_CURSOR_OFF);
    Lcd_Custom_Cmd(LCD_CLEAR);
    Lcd_Custom_Out(1, 1, "DataLoggerV2");
}

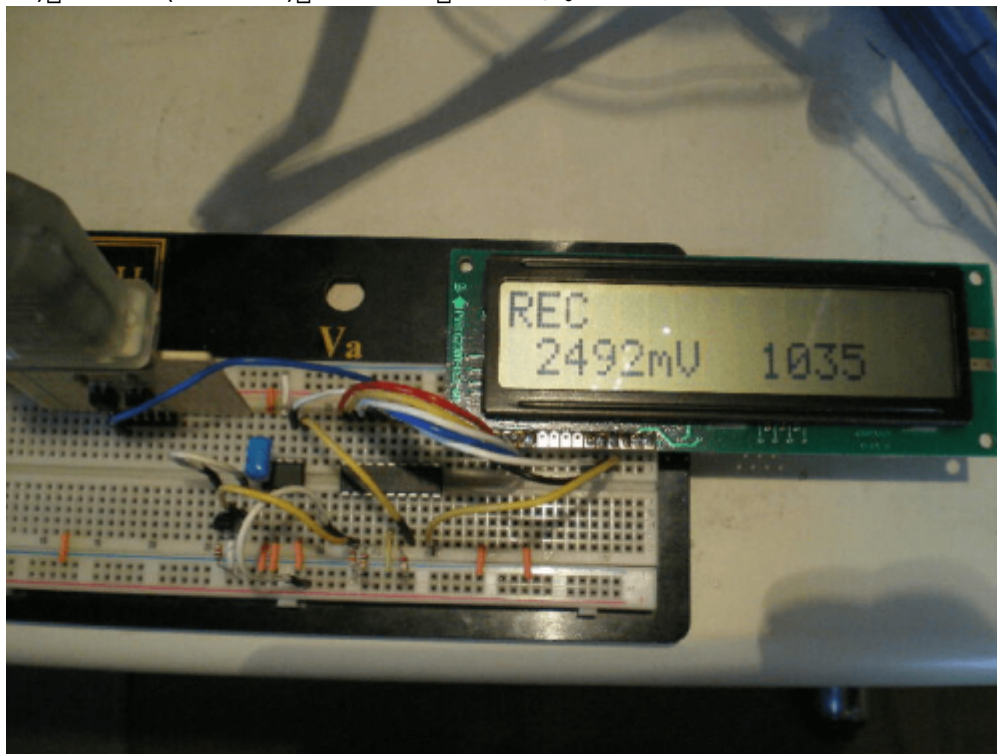
```

```
//RS232Cを初期化する。
Usart_Init(9600);
Usart_Write_String("\r\nDataLoggerV2\r\n");
//EEPROMを初期化する。
EEPROM_24C256_Init();
// CCPを設定する。
PIE1.CCP1IE = 1;
PIR1.CCP1IF = 0;
CCP1CON.CCP1M3 = 1;           //trigger special event mode
CCP1CON.CCP1M2 = 0;
CCP1CON.CCP1M1 = 1;
CCP1CON.CCP1M0 = 1;
CCPR1L = 0xA8;               // 0.1sec...10hz...クロックが8Mhzの時
CCPR1H = 0x61;               // 0.1sec...(1÷8000000)*4*8*25000
// TIMER1を設定する。
PIE1.TMR1IE = 0;
PIR1.TMR1IF = 0;
TMR1L = 0;
TMR1H = 0;
T1CON.T1CKPS0 = 1;
T1CON.T1CKPS1 = 1;
T1CON.TMR1ON = 1;
//変数を初期化する。
TimerCnt = 0;
swRec = OFF;
swPlay = OFF;
swRecCnt = 0;
swPlayCnt = 0;
// 割り込み(全体)の設定
INTCON.PEIE = 1;
INTCON.GIE = 1;
//
while (1) {
    //記録スイッチが押されていれば、アナログデータの記録を行う。
    if (swRec == ON) {
        swRec = OFF;
        Lcd_Custom_Cmd(LCD_CLEAR);
        Lcd_Custom_Out(1, 1, "REC ");
        Data_Rec();
        Lcd_Custom_Cmd(LCD_CLEAR);
        Lcd_Custom_Out(1, 1, "DataLoggerV2");
    }
    //再生スイッチが押されていれば、アナログデータの再生を行う。
    if (swPlay == ON) {
        swPlay = OFF;
        Lcd_Custom_Cmd(LCD_CLEAR);
        Lcd_Custom_Out(1, 1, "PLAY");
        Data_Play();
        Lcd_Custom_Cmd(LCD_CLEAR);
        Lcd_Custom_Out(1, 1, "DataLoggerV2");
    }
}
```

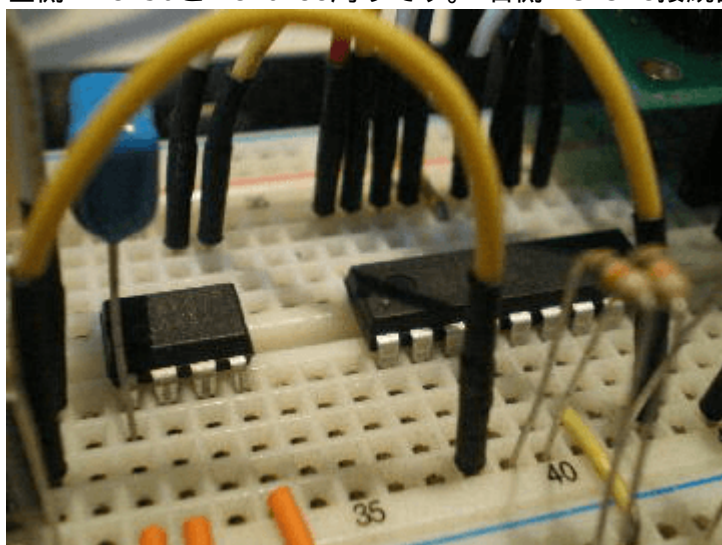
```
}  
}  
  
//*****  
*
```

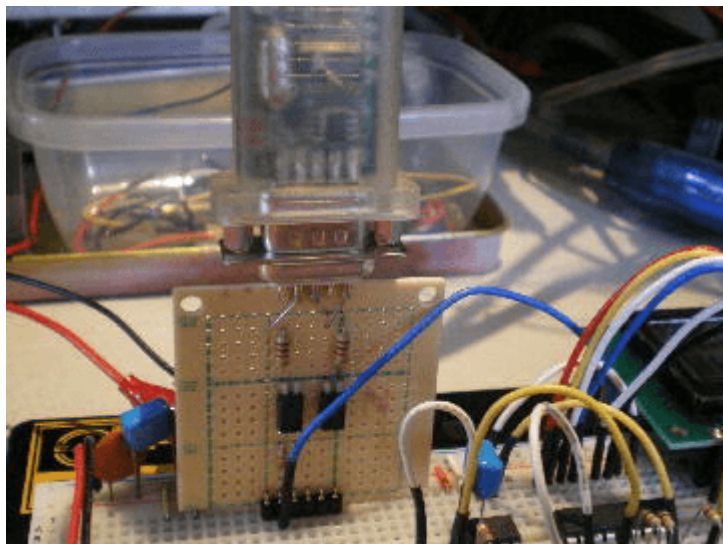
動作確認

いつものブレッドボードで確認しました。左側からRS232C接続部TL431(高精度シャント・レギュレータ)24C256(EEPROM)PIC16F88LCDです。



左側:24C256とPIC16F88周りです。 右側:RS232C接続部周りです。

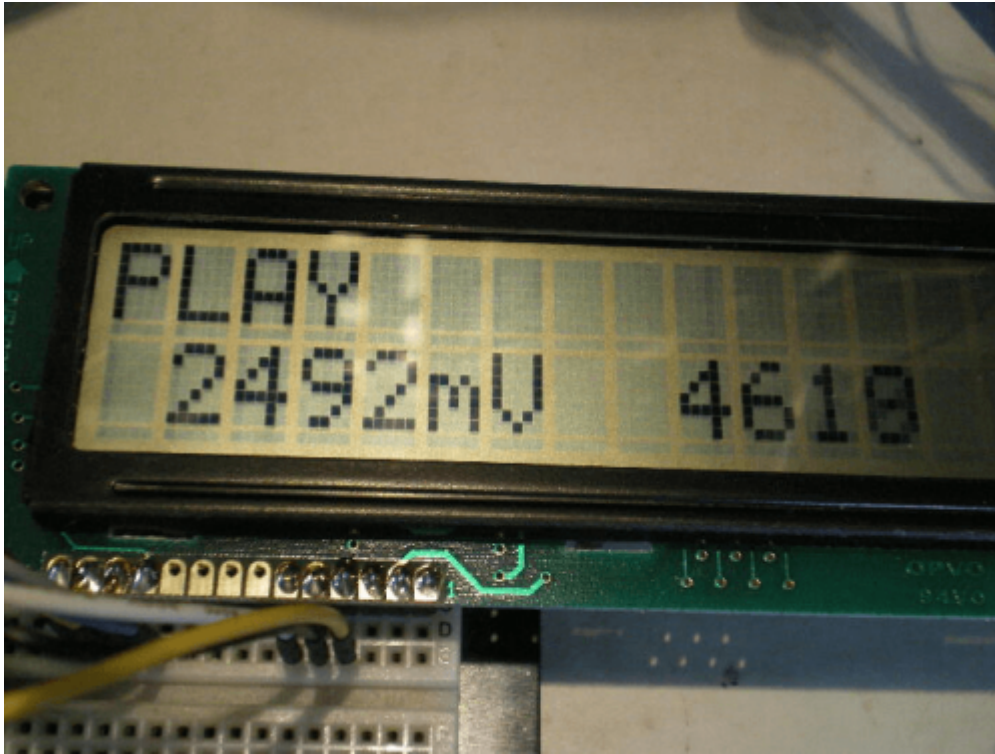




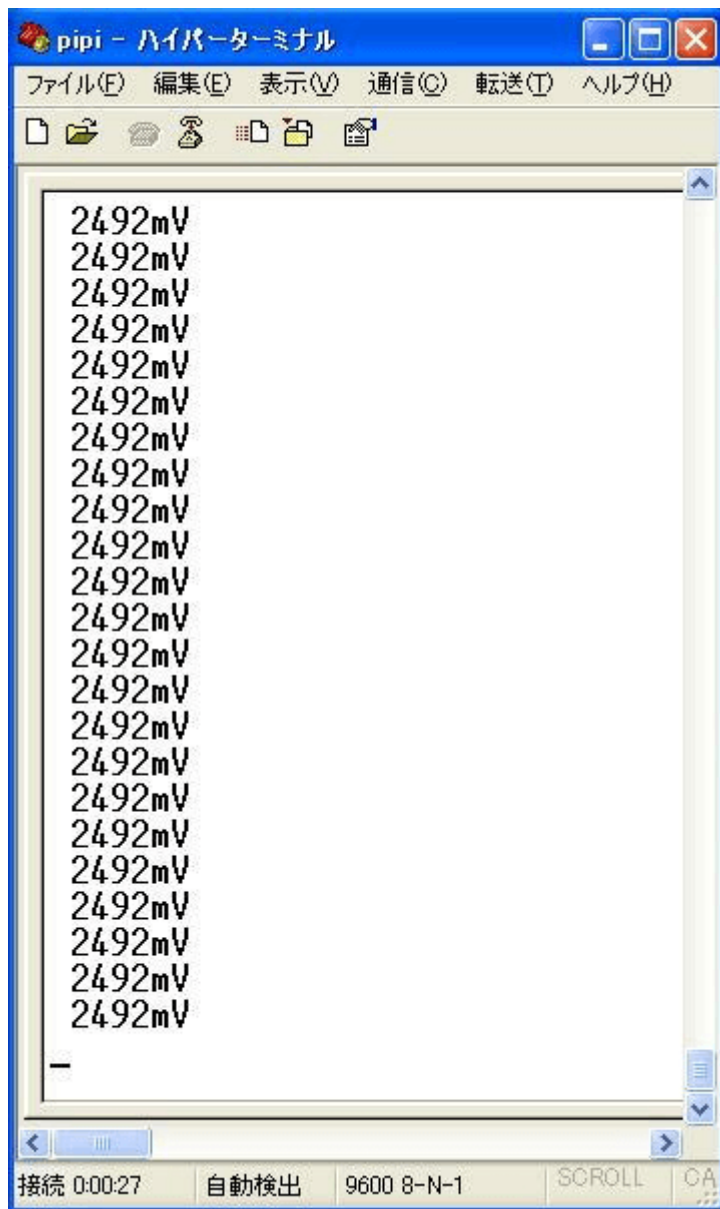
記録時のLCD表示です。記録を途中で中断したい場合には、再度記録スイッチを押すと中断します。左下:A/D変換で取り込んだ電圧の表示です。右下:記録回数(1秒周期、1分周期の設定が可能です)。



再生時のLCD表示です。再生を途中で中断したい場合には、再度再生スイッチを押すと中断します。左下:A/D変換で取り込んだ電圧の表示です。右下:記録回数です。(高速に変化します)



再生時のパソコンの通信ソフト(ハイパーターミナル)の表示です。このデータを表計算ソフト(Excel)で取り込んでグラフ表示させると電圧の変化が視覚的にわかります。



機能強化案

EEPROM(24C256)は、アドレス指定(A0□A1)が出来ますので、後3個追加することが出来ます。これによって、1秒周期(記録時間:約18時間)または1分周期(記録時間:約44日)とすることが出来ます。

EEPROM(24LC256)は、アドレス指定(A0□A1□A2)が出来ますので、後7個追加することが出来ます。これによって、1秒周期(記録時間:約36時間)または1分周期(記録時間:約88日)とすることが出来ます。

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:pic16f88:91>

Last update: **2020/05/01 17:53**

