

10チャンネルデータロガー(PIC18F2320)

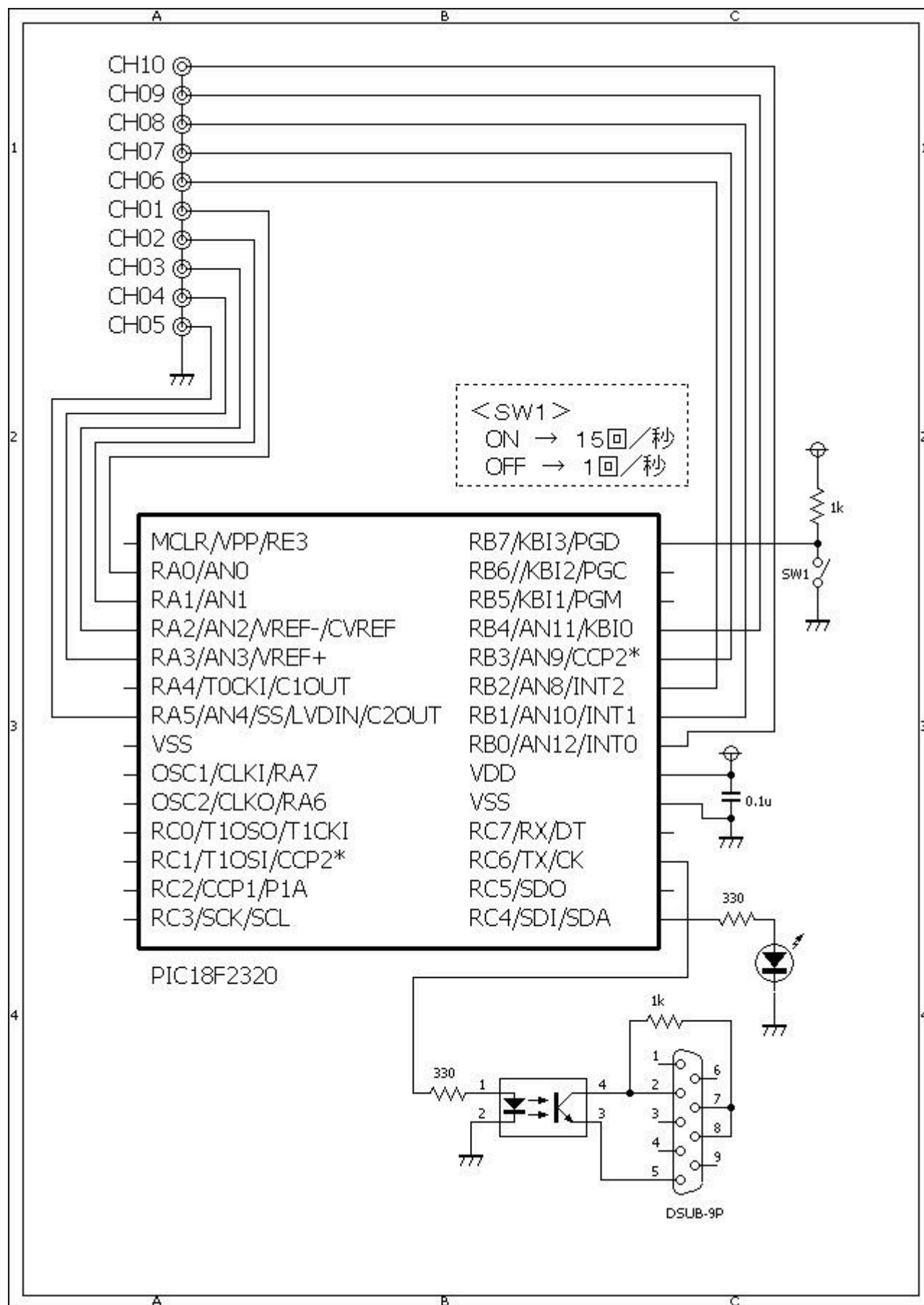
概要

初めてPIC18シリーズを使用しましたPIC18F2320を使用して、単純にAD変換したデータをRS232C経由でPCに送るだけのものを作成しました。尚、計測モードは次の2種類としました。

- 1秒間に1回、AD変換(10チャンネル分)してPCにデータを送る。
- 1秒間に15回、AD変換(10チャンネル分)してPCにデータを送る。

AD変換の精度は10ビットです。なので最小単位は約5mvとなります。精度を高めるには、電源電圧VCCを $1023 \times 5\text{mv} = 5115\text{mv}$ と安定化する必要があります。

回路図



ソースコード

DataLogger_10channel.c

```
//*****
*
/*
「10チャンネルデータロガー」

・10チャンネルのアナログデータを10bit変換し、10bitで送信し
ます。（秒間約5回のサンプリング）

*/
//*****
*

#define LED          PORTC.F4
#define SW           PORTB.F7

//*****
*

static unsigned short startCount;

//*****
*

void interrupt(){
    if (PIR1.TMR1IF == 1) {
        PIR1.TMR1IF = 0;
        //
        LED = ~LED;
        //
        if (startCount > 0)
            startCount--;
    }
}

//*****
*

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

```
//*****
*

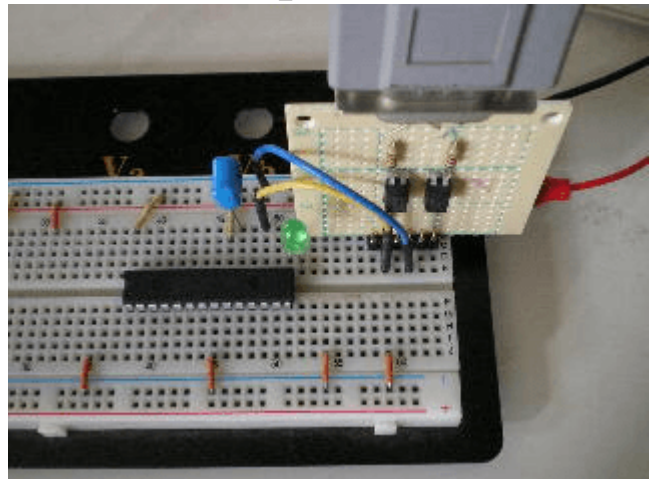
void main()
{
    static unsigned int    ad0, ad1, ad2, ad3, ad4, ad8, ad9,
ad10, ad11, ad12, cnt;
    static unsigned char   buf[10];
    //
    OSCCON.IRCF2=1;
    OSCCON.IRCF1=1;
    OSCCON.IRCF0=1;
    //
    TRISA = 0b11111111;
    TRISB = 0b11111111;
    TRISC = 0b00000000;
    //
    // TIMER1の設定
    PIE1.TMR1IE = 1;
    PIR1.TMR1IF = 0;
    TMR1L = 0;
    TMR1H = 0;
    T1CON.T1CKPS0 = 1;
    T1CON.T1CKPS1 = 1;
    T1CON.TMR1ON = 1;
    //
    startCount = 4;
    cnt = 0;
    //
    Usart_Init(9600);
    Delay_ms(100);
    Usart_Write_String("DataLogger(10channel)\r\n");
    // 割り込み(全体)の設定
    INTCON.PEIE = 1;
    INTCON.GIE = 1;
    //
    while(1) {
        if ((SW == 1) && (startCount != 0))
            continue;
        //
        cnt++;
        WordToStr(cnt, buf);
        Usart_Write_String(buf);
        Usart_Write_String(": ");
        ad0 = Adc_Read(0);
        ad1 = Adc_Read(1);
        ad2 = Adc_Read(2);
        ad3 = Adc_Read(3);
        ad4 = Adc_Read(4);
        ad8 = Adc_Read(8);
    }
}
```

```
    ad9 = Adc_Read(9);
    ad10 = Adc_Read(10);
    ad11 = Adc_Read(11);
    ad12 = Adc_Read(12);
    WordToStr(ad0 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad1 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad2 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad3 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad4 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad8 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad9 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad10 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad11 * 5, buf);
    Usart_Write_String(buf);
    WordToStr(ad12 * 5, buf);
    Usart_Write_String(buf);
    Usart_Write_String("\r\n");
    startCount = 4;
}
}
```

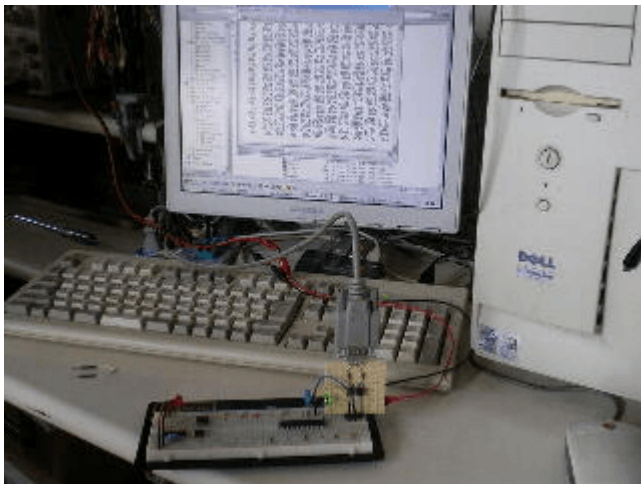
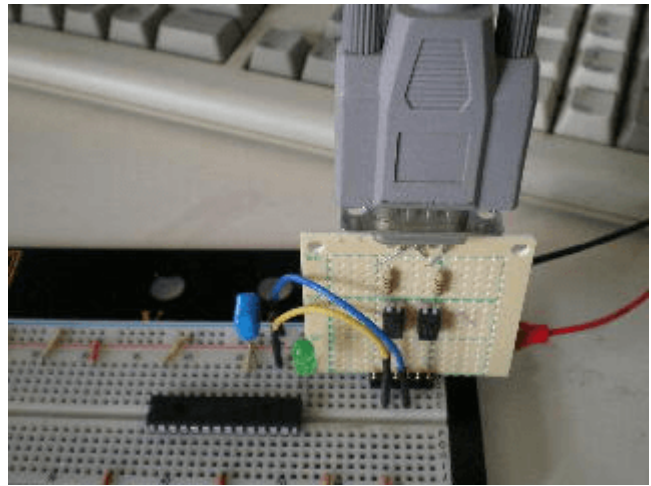
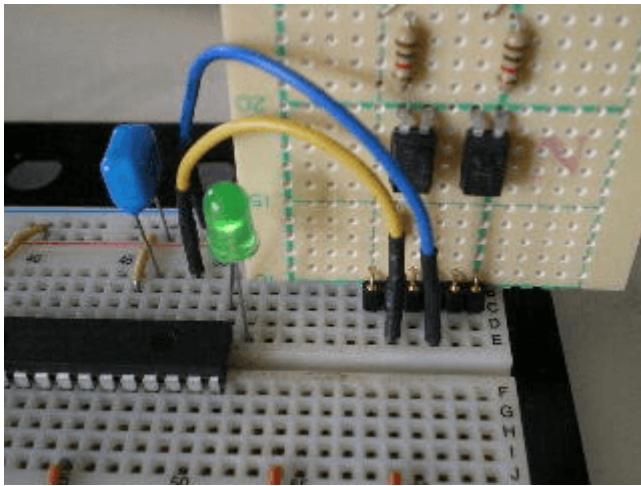
//*****
*

動作確認

今回は部品点数がとて少ないのでブレッドボードに組んで確認しました。PCとの接続には以前に作成



したRS232Cレベル変換ユニットを使用しています。



Windowsに標準で付いているターミナルソフト(ハイパーターミナル)を使用してデータを表示します。一番左のデータは、単なるカウント値です。その右からCH1~CH10の順番です。

ping - ハイパーターミナル

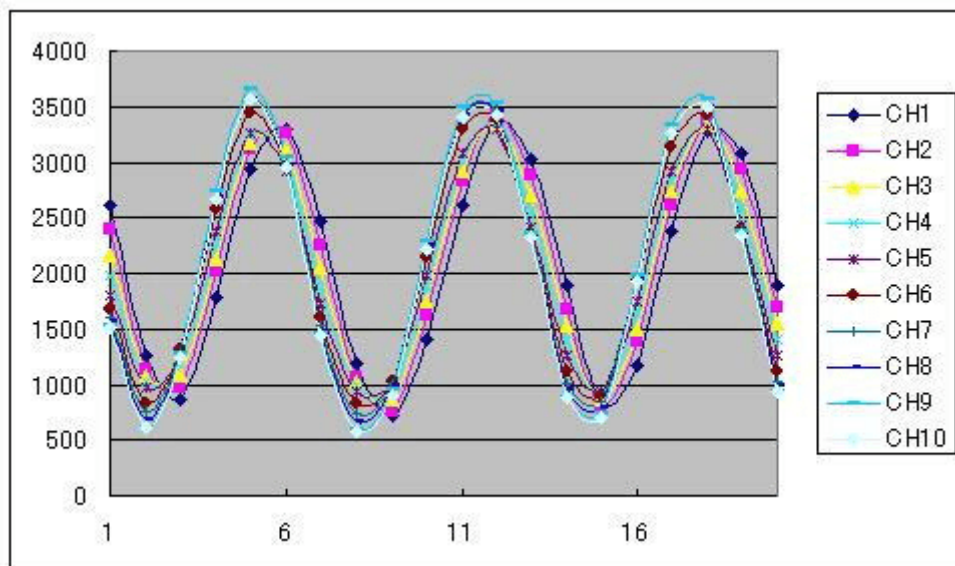
ファイル(F) 編集(E) 表示(V) 通信(C) 転送(T) ヘルプ(H)

Datalogger(10channel)

1:	1695	1530	1345	1185	1050	940	895	865	845	860
2:	800	755	800	790	790	765	715	675	630	610
3:	990	1140	1225	1330	1410	1535	1580	1570	1585	1525
4:	2020	2255	2385	2505	2580	2780	2890	2890	2970	2880
5:	3060	3215	3245	3290	3305	3485	3605	3600	3685	3580
6:	3260	3200	3045	2950	2855	2910	2950	2940	2985	2925
7:	2345	2120	1885	1695	1530	1380	1300	1230	1210	1190
8:	1000	890	910	905	875	780	695	650	570	555
9:	935	1105	1245	1385	1485	1565	1585	1590	1600	1520
10:	1980	2220	2345	2460	2530	2750	2865	2880	2930	2840
11:	3030	3200	3210	3255	3275	3475	3605	3595	3705	3600
12:	3260	3185	3040	2920	2795	2830	2870	2860	2910	2845
13:	2335	2130	1925	1735	1590	1455	1390	1335	1310	1285
14:	1050	940	940	915	890	780	690	635	585	565
15:	760	830	935	1050	1120	1135	1130	1115	1115	1085
16:	1610	1840	1955	2090	2185	2375	2490	2505	2565	2475
17:	2800	2995	3080	3150	3200	3385	3510	3530	3615	3525
18:	3345	3355	3275	3225	3135	3220	3265	3245	3305	3220
19:	2750	2585	2365	2195	2060	1975	1960	1910	1925	1885
20:	1525	1350	1265	1175	1085	950	875	830	780	760
21:	705	680	730	775	810	760	700	665	615	605
22:	1070	1275	1420	1560	1665	1815	1885	1880	1905	1840

接続 02915 自動検出 9600 8-N-1 SCROLL CAPS NUM キャ* エコーを印

取り込んだデータをExcelを使用してグラフ表示させてみました。入力端子をオープン状態にして計測しましたのでノイズ(商用の60Hz)が乗っています。入力端子を100kΩ位の抵抗でプルダウンすると良い



と思います。

From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:otherpic:159&rev=1588234308>

Last update: 2025/10/17 14:27

