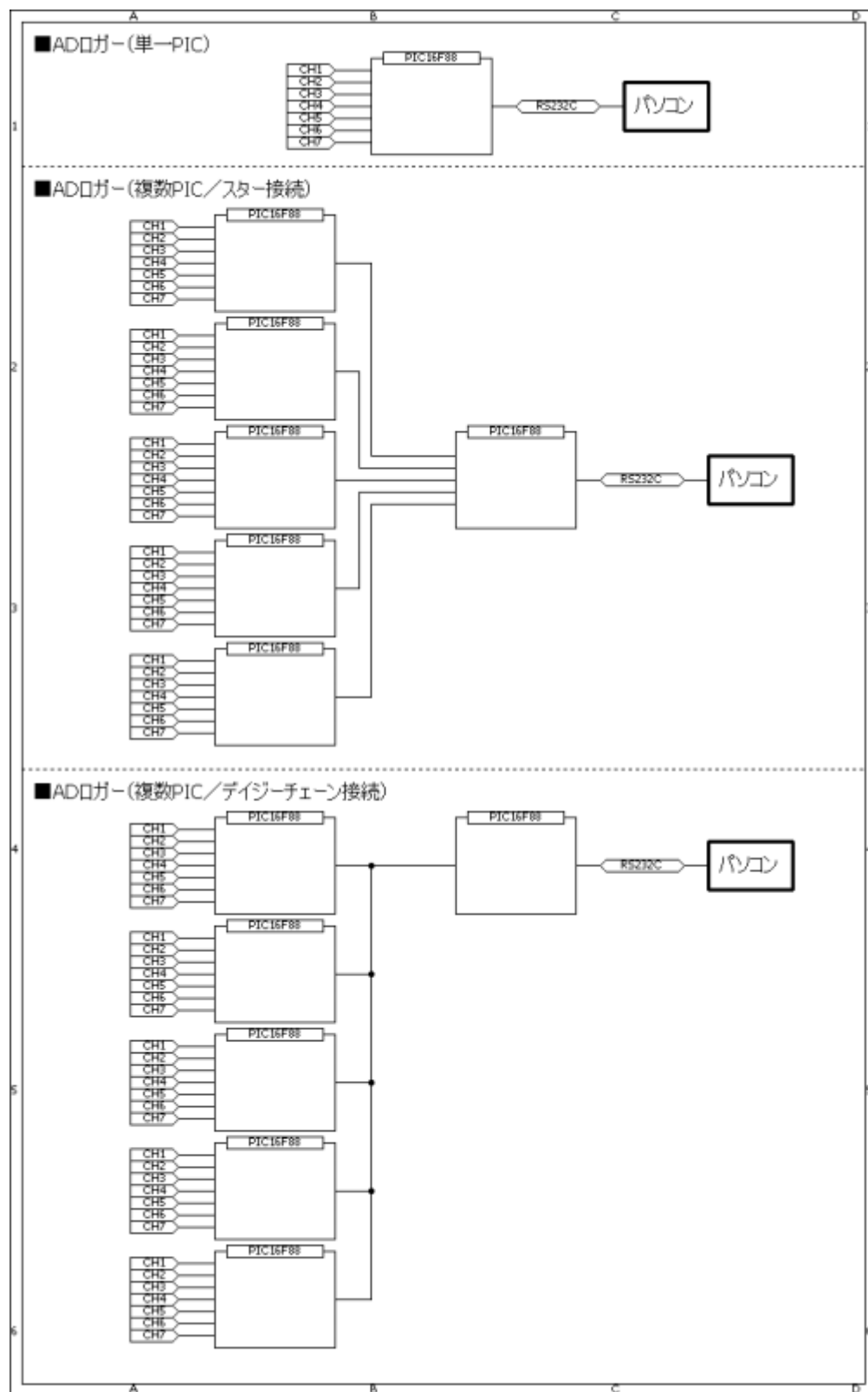


簡易多チャンネルロガー(複数PIC/35チャンネル)

概要

以前に製作した、アナログデータロガー(複数PIC使用)は、多チャンネルロガーを複数のPICをダイジーチェーンで接続する方式で実現しました。

今回は、1個のPICからシリアル通信(ソフトウェア方式)が複数可能かどうかを確認する意味も含めて、スター接続する方式で動作確認してみました。



動作原理

1個のマスタPICに5個のスレーブPICをスター接続し、最大35チャンネルのアナログデータを測定し、その結果を、マスターに接続したパソコンにデータ転送(9600bps)します。

動作原理(ハードウェア)

マスタPIC

- パソコンとの接続
PIC内蔵のUSARTモジュールを使用して、パソコンとの通信を行います(9600bps、8ビット、ノンパリティ)
- スレーブPICとの接続
I/Oポートを使用して、ソフトウェア的にUSART通信を行います(9600bps、8ビット、ノンパリティ)
スレーブPICを5個接続するために、10個のI/Oポートを使用します(Rx1~Rx5、Tx1~Tx5)

スレーブPIC

- アナログ入力
内蔵のA/D変換モジュールのAN0~AN6までの、7チャンネルを使用します。
入力電圧範囲は、0V~5Vで、分解能は、約4.9mVになります。
- マスタPICとの接続
PIC内蔵のUSARTモジュールを使用して、マスタPICとの通信を行います(9600bps、8ビット、ノンパリティ)

動作原理(ソフトウェア)

マスターモードとスレーブモードの切り替え

- 起動時に、スイッチ(SW_MODE)をオンにするとマスターモードとなり、オフにするとスレーブモードとなります。

マスター処理

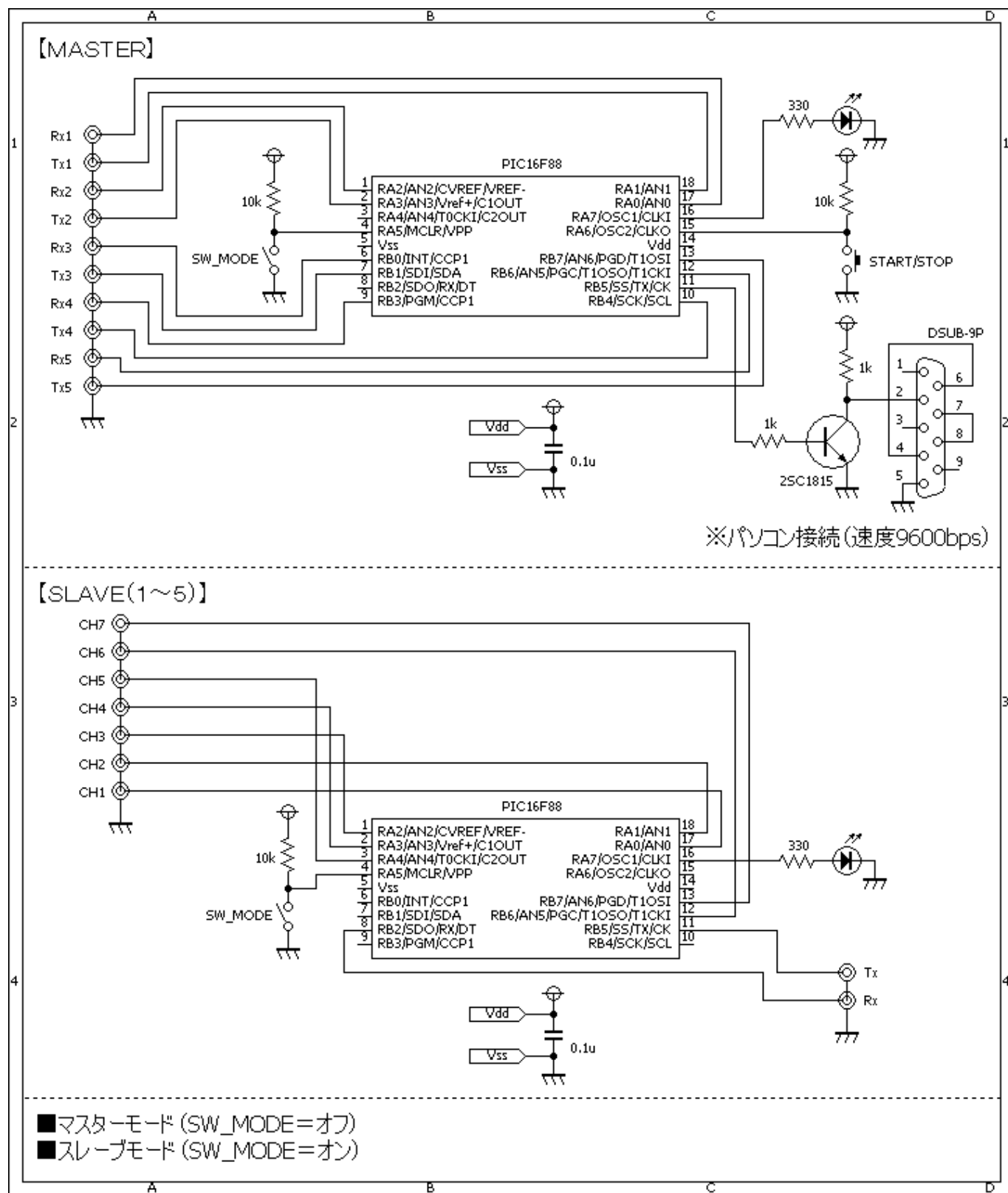
- パソコン接続用にUSARTモジュールを初期化します。
- アナログチャンネルを全て未使用に設定します。
- スイッチ(SW_START_STOP)が押下されるのを待ちます。
- スレーブ接続用のポートをシリアルポートとして初期化します。
- スレーブ(1)にデータ'!'を送信します。
- スレーブ(1)からのデータを受信します。(データの終端は、0x00です)
- 受信したデータをパソコン側にデータ転送します。
- スレーブ(2)~(5)も、(1)同様の処理を行います。

接続スレーブ	使用ポート	
1	Rx	PORTA.B0
	Tx	PORTA.B1
2	Rx	PORTA.B2
	Tx	PORTA.B3
3	Rx	PORTB.B0
	Tx	PORTB.B1
4	Rx	PORTB.B3
	Tx	PORTB.B4
5	Rx	PORTB.B6
	Tx	PORTB.B7

スレーブ処理

- マスター接続用にUSARTモジュールを初期化します。
- アナログチャネルを全て使用に設定します。
- マスターからのデータ「!」を受信します。
- 全てのアナログチャネルのデータをA/D変換して取り込み、電圧値に換算し、文字列変換します。
- 文字列変換されたデータをマスターに送信します。

回路図



ソースコード

[ad_logger_multi_pic_v2.c](#)

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

< 簡易多チャネルロガー (複数PIC 35チャネル) >

```

*/
//*****
*
sbit  SW_MODE      at  RA5_bit;
sbit  SW_START_STOP at  RA6_bit;
sbit  LED          at  RA7_bit;
#define CR  0x0D
#define LF  0x0A
//*****
*
extern void  main();
extern void  master();
extern void  slave();
extern void  UART1_Write_Text_Ex(char * UART_text);
extern void  collection(char *msg);
//*****
*
char  error, rdt, sdt, buf[40];
//*****
*
// ■ メイン関数
void  main()
{
    short  cnt;
    //
    OSCCON = 0b01110000;
    TRISA  = 0b01111111;
    TRISB  = 0b11111111;
    //
    UART1_Init(9600);
    if (SW_MODE == 1) {
        ANSEL = 0b00000000;
    } else {
        ANSEL = 0b01111111;
        ADC_Init();
    }
    //
    for (cnt = 0; cnt < 10; cnt++) {
        LED = 1;
        Delay_ms(50);
        LED = 0;
        Delay_ms(50);
    }
    //
    while (1) {
        if (SW_MODE == 1) {
            master();
        } else {
            slave();
        }
    }
}

```

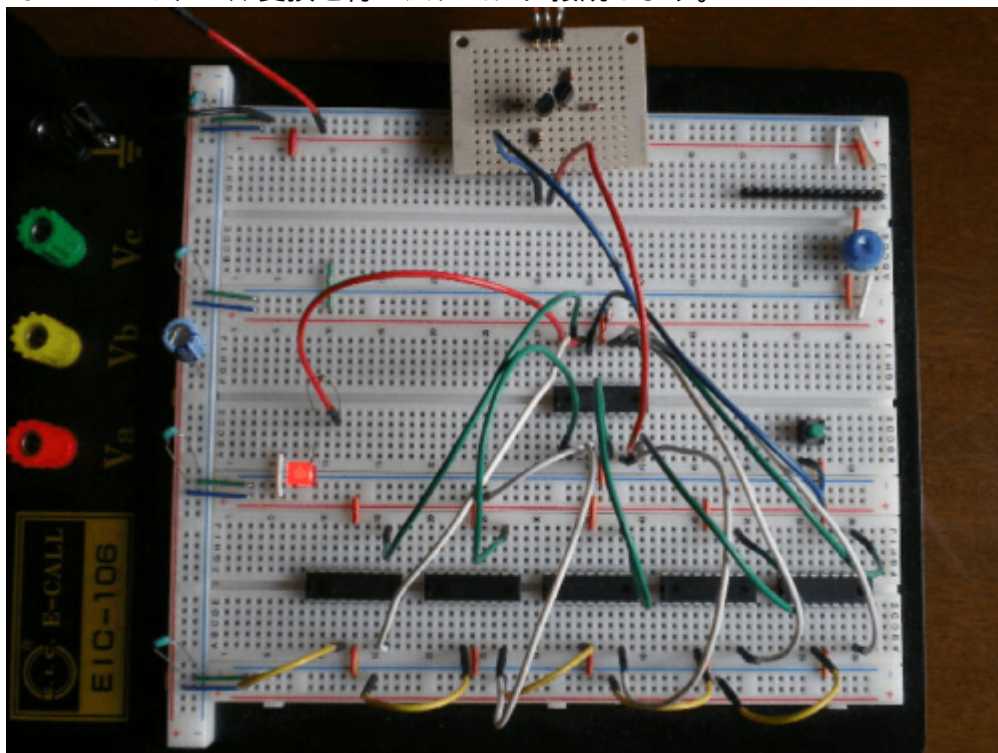
```
    }  
}  
//*****  
*  
void    master()  
{  
    short    mode;  
    //  
    mode = 0;  
    while (1) {  
        if ((SW_START_STOP == 0) && (mode == 0)) {  
            mode = 1;  
            LED = 1;  
            while (SW_START_STOP == 0) {  
                Delay_ms(100);  
            }  
        }  
        //  
        if ((SW_START_STOP == 0) && (mode == 1)) {  
            mode = 0;  
            LED = 0;  
            while (SW_START_STOP == 0) {  
                Delay_ms(100);  
            }  
        }  
        //  
        if (mode == 0) {  
            continue;  
        }  
        //  
        error = Soft_UART_Init(&PORTA, 0, 1, 9600, 0);  
        collection(buf);  
        UART1_Write_Text(buf);  
        //  
        error = Soft_UART_Init(&PORTA, 2, 3, 9600, 0);  
        collection(buf);  
        UART1_Write_Text(buf);  
        //  
        error = Soft_UART_Init(&PORTB, 0, 1, 9600, 0);  
        collection(buf);  
        UART1_Write_Text(buf);  
        //  
        error = Soft_UART_Init(&PORTB, 3, 4, 9600, 0);  
        collection(buf);  
        UART1_Write_Text(buf);  
        //  
        error = Soft_UART_Init(&PORTB, 6, 7, 9600, 0);  
        collection(buf);  
        UART1_Write_Text(buf);  
        UART1_Write_Text("\r\n");  
    }  
}
```

```
}
//*****
*
void    slave()
{
    short    cnt;
    int      ad;
    //
    while (1) {
        if (UART1_Data_Ready() != 1) {
            continue;
        }
        rdt = UART1_Read();
        if (rdt != '!') {
            continue;
        }
        LED = 1;
        for (cnt = 0; cnt < 7; cnt++) {
            ad = ADC_Get_Sample(cnt);
            ad *= 4.8828125;
            WordToStr(ad, &buf[cnt * 5]);
        }
        LED = 0;
        buf[35] = 0x00;
        UART1_Write_Text_Ex(buf);
    }
}
//*****
*
void    UART1_Write_Text_Ex(char * UART_text)
{
    while (*UART_text != 0x00) {
        UART1_Write(*UART_text);
        UART_text++;
        Delay_ms(10);
    }
    UART1_Write(0x00);
}
//*****
*
void    collection(char *msg)
{
    short    cnt;
    //
    Soft_UART_Write('!');
    cnt = 0;
    while (1) {
        msg[cnt] = Soft_UART_Read(&error);
        if (msg[cnt] == 0x00) {
            break;
        }
    }
}
```

```
cnt++;  
}  
}  
//*****  
*
```

動作確認

中央のPICがマスターです。下部の5個のPICがスレーブです。上側の小さな基板(トランジスタ+抵抗)でRS232Cのレベル変換を行いパソコンに接続します。



Windows標準の通信ソフト(ハイパーターミナル)でデータを受信します。1件分のデータが長いので、ハイパーターミナル上は、1件3行で表示されます。テキストキャプチャしたログファイル上は、1件1

9600bps - 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)

1020 991 874 776 722 805 756 1118 1093 927 903 844 913 893 952 932
 820 756 683 761 693 981 952 825 776 722 810 747 1259 1225 1088 1040
 971 1010 1030

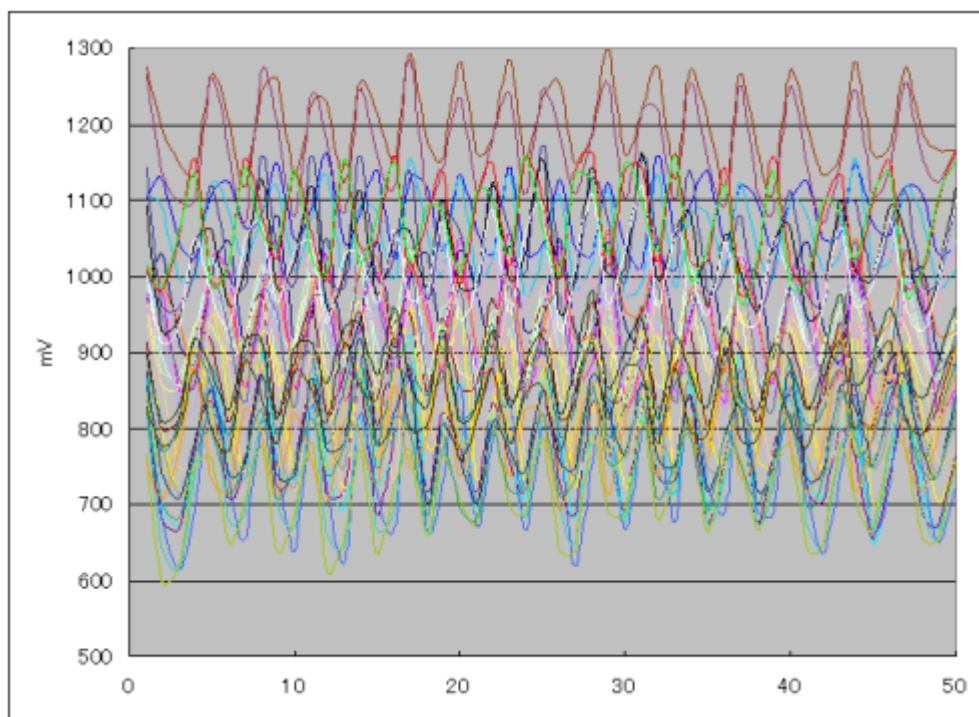
1件分のデータ

942 864 717 668 654 795 776 1074 1020 854 864 839 942 922 878 810
 659 654 629 756 732 922 878 712 698 683 810 766 1210 1157 1132 1069
 952 1035 1054
 893 849 732 717 727 883 859 1025 976 844 859 844 976 1162 1118 981 976
 639 649 644 805 781 883 830 693 703 678 844 815 1162 1118 981 976
 966 1079 1098
 908 883 791 791 791 927 908 1010 1000 903 922 917 1035 1005 820 751
 693 732 727 888 849 888 864 756 781 761 913 878 1157 1132 1030 1069
 1064 1157 1137
 996 1000 922 869 815 893 839 1093 1079 976 952 903 966 917 932 942
 849 839 776 869 781 961 952 849 839 791 888 810 1240 1250 1123 1108
 1044 1054 1030
 1015 971 844 737 698 805 761 1098 1059 898 859 800 878 844 952 922
 786 722 649 742 678 986 957 810 761 698 810 751 1245 1225 1074 1010
 976 1010 1025
 917 844 703 659 668 810 795 1015 947 800 795 786 898 883 864 786
 634 620 625 756 732 927 888 727 712 693 825 776 1201 1152 991 981
 957 1030 1054

接続 15812 自動検出 9600 8-N-1 SCROLL CAPS NUM キー エコーも印

行になります。

受信したデータを、Excelでグラフ表示させて見ました。



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://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:152>

Last update: **2020/05/01 18:42**

