

SDカード ドReader&Writer(RS232C)(PIC18F4520)

概要

PICを使った製作物では、パソコンとのやりとりには、よくRS232Cを使います。その時には、図Aのような構成が一般的ですが、回路構成や操作が煩雑になります。

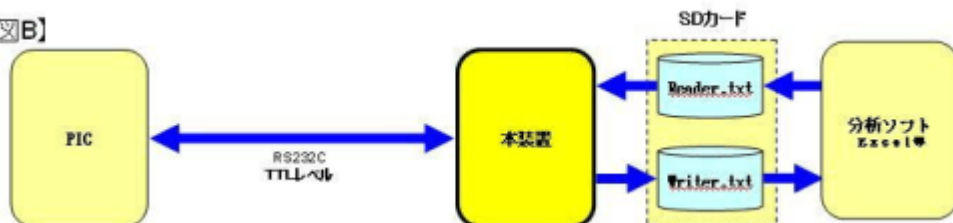
そこで図BのようにRS232Cのインターフェイス回路や通信ソフトを省略し、PICからのRS232C(TTLレベル)の入出力(ファイルへの書き込み読み込み)を出来るだけ単純にした本装置を製作しました。

本装置を1台手元に用意しておくことで、データの蓄積や設定が容易になるので、電子工作の応用範囲が、大いに広がるのではないのでしょうか。

【図A】



【図B】



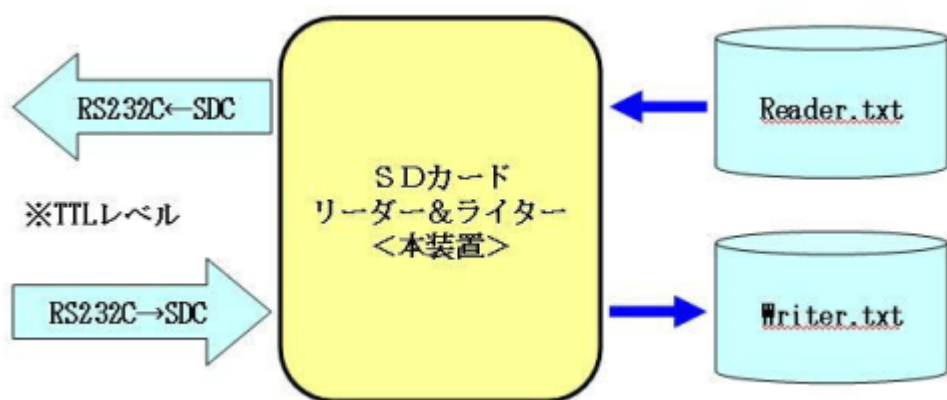
動作原理

RS232Cインターフェイス(TTLレベル)を使用した、本装置からのデータの送信では、

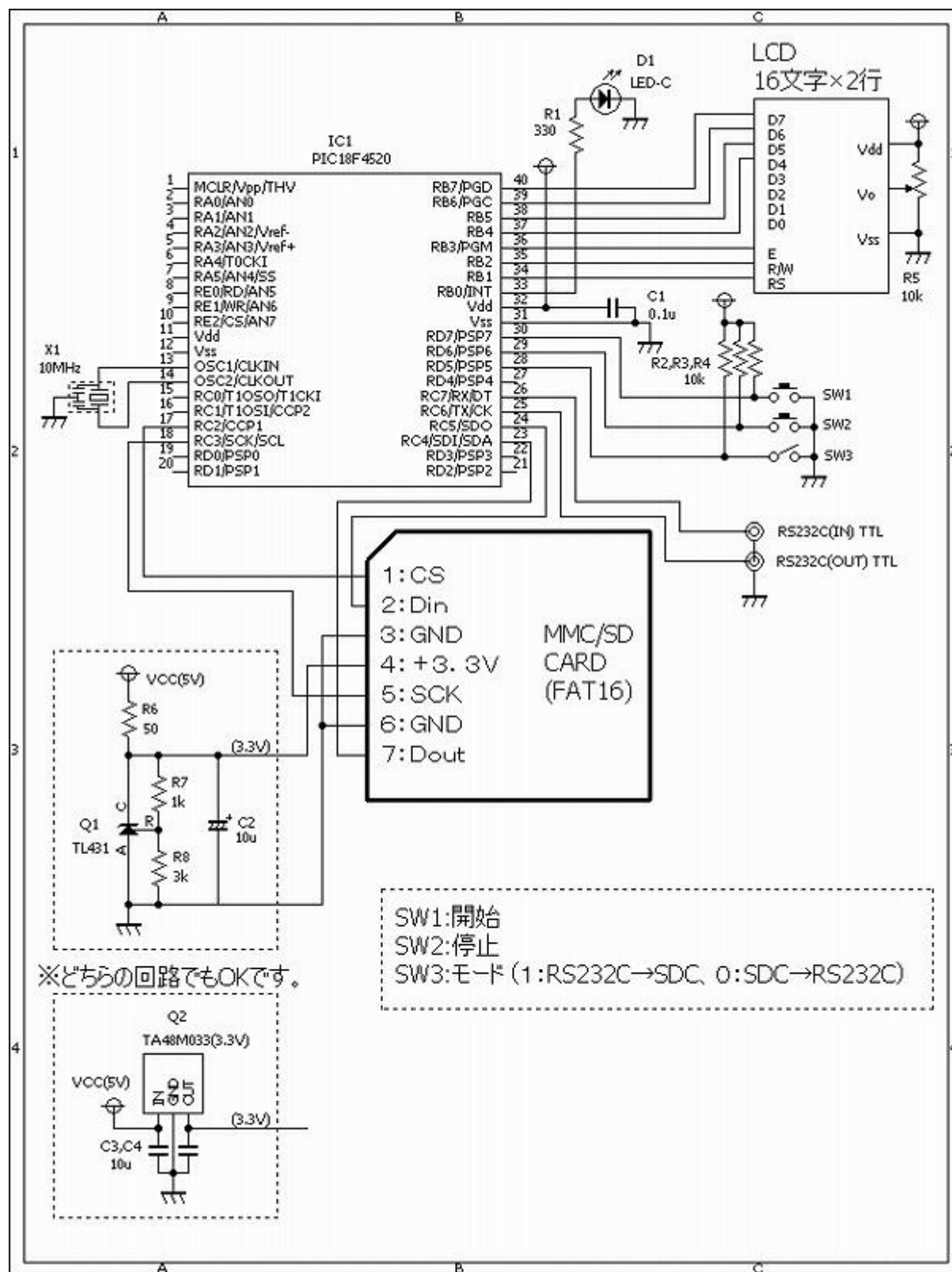
1. 開始スイッチが押されると、
2. SDカードにあるファイル“reader.txt”をオープンする。
3. ファイルからデータを、1バイト読み込んでRS232Cへ出力する。
4. 読み込んだデータの件数を、LCDへ表示する。
5. ファイルの最後まで、3.4.を繰り返す。

RS232Cからの、本装置への受信では、

1. 開始スイッチが押されると、
2. SDカードにあるファイル“writer.txt”をオープンする。
3. RS232Cからデータを読み込み、メモリバッファに保存する。
4. 保存したデータが128バイトになったら、ファイルへ書き込む。
5. 停止スイッチが押されるまで、3.4.を繰り返す。



回路図



ソースコード

SdcReaderWriter.c

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

```

□□□□□□□□↔□□カ-ト□□□□□□□□□□□□
*/
//*****
*

#define      LED      PORTB.F0

#define      SW_START  PORTD.F7
#define      SW_STOP   PORTD.F6
#define      SW_MODE   PORTD.F5

#define      ON        1
#define      OFF       0

#define      CR        0x0d
#define      LF        0x0a

#define      DATA_SIZE 128

unsigned short flg;
unsigned short msg1[DATA_SIZE], len1;
unsigned short msg2[DATA_SIZE], len2;
unsigned short msg3[DATA_SIZE], len3;

//*****
*

static char    buf[32];

//*****
*

void interrupt()
{
    while (Usart_Data_Ready()) {
        switch (flg) {
            case 0:
                msg1[len1] = Usart_Read();
                len1++;
                if (len1 == DATA_SIZE) {
                    flg = 1;
                }
                break;
            case 1:
                msg2[len2] = Usart_Read();
                len2++;
                if (len2 == DATA_SIZE) {
                    flg = 2;
                }
                break;
        }
    }
}
```

```
        case 2:
            msg3[len3] = Usart_Read();
            len3++;
            if (len3 == DATA_SIZE) {
                flg = 0;
            }
            break;
        }
    }
} //~

//*****
*

void Usart_Write_Str(char *str)
{
    unsigned short i;

    i = 0;
    while (str[i] != 0x00) {
        USART_Write(str[i]);
        i++;
    }
    USART_Write(CR);
    USART_Write(LF);
} //~

//*****
*

void SDCwriter()
{
    //変数の定義
    short cnt;
    long dataSize;
    //変数の初期化
    flg = 0;
    len1 = 0;
    len2 = 0;
    len3 = 0;
    dataSize = 0;
    //
    Lcd_Custom_Out(1, 1, "SDC-Writer");
    for (cnt = 0; cnt < 10; cnt++) {
        LED = ON;
        Delay_ms(50);
        LED = OFF;
        Delay_ms(50);
    }
    //ファイルの作成
    Mmc_Fat_Assign("writer.txt", 0xA0);
}
```

```
Mmc_Fat_Rewrite();
// Mmc_Fat_Write("$START\r\n", 8);
// 割り込みを許可する。
PIE1.RCIE = 1;
PIR1.RCIF = 0;
INTCON.PEIE = 1;
INTCON.GIE = 1;
//停止スイッチが押されるまで処理を繰り返す。
while (SW_STOP == 1) {
    if (len1 == DATA_SIZE) {
        LED = ON;
        Mmc_Fat_Write(msg1, len1);
        LED = OFF;
        len1 = 0;
        //
        dataSize += DATA_SIZE;
    }
    if (len2 == DATA_SIZE) {
        LED = ON;
        Mmc_Fat_Write(msg2, len2);
        LED = OFF;
        len2 = 0;
        //
        dataSize += DATA_SIZE;
    }
    if (len3 == DATA_SIZE) {
        LED = ON;
        Mmc_Fat_Write(msg3, len3);
        LED = OFF;
        len3 = 0;
        //
        dataSize += DATA_SIZE;
    }
    //書き込んだ件数を□□□に表示する。
    LongToStr(dataSize, buf);
    Lcd_Custom_Out(2, 1, buf);
}
// 割り込みを禁止する。
PIE1.RCIE = 0;
PIR1.RCIF = 0;
INTCON.PEIE = 0;
INTCON.GIE = 0;
//
switch (flg) {
case 0:
    LED = ON;
    Mmc_Fat_Write(msg1, len1);
    dataSize = dataSize + len1;
    LED = OFF;
    break;
```

```
    case 1:
        LED = ON;
        Mmc_Fat_Write(msg2, len2);
        dataSize = dataSize + len2;
        LED = OFF;
        break;
    case 2:
        LED = ON;
        Mmc_Fat_Write(msg3, len3);
        dataSize = dataSize + len3;
        LED = OFF;
        break;
}
//
// Mmc_Fat_Write("$STOP\r\n", 7);
// 書き込んだ件数を□□□に表示する。
LongToStr(dataSize, buf);
Lcd_Custom_Out(2, 1, buf);
//
for (cnt = 0; cnt < 10; cnt++) {
    LED = ON;
    Delay_ms(50);
    LED = OFF;
    Delay_ms(50);
}
Lcd_Custom_Out(1, 1, "          ");
}

//*****
*

void SDCreader()
{
    //変数の定義
    unsigned long    fsize, i;
    unsigned short   character, cnt;
    //
    Lcd_Custom_Out(1, 1, "SDC-Reader");
    for (cnt = 0; cnt < 10; cnt++) {
        LED = ON;
        Delay_ms(50);
        LED = OFF;
        Delay_ms(50);
    }
    //
    Mmc_Fat_Assign("reader.txt", 0);
    fsize = Mmc_Fat_Get_File_Size();
    // Mmc_Fat_Reset(&fsize);
    for (i = 1; i <= fsize; i++) {
        Mmc_Fat_Read(&character);
        Usart_Write(character);
    }
}
```

```
//読み込んだ件数を□□□に表示する。
LongToStr(i, buf);
Lcd_Custom_Out(2, 1, buf);
//
if (SW_STOP == 0)
    break;
}
//
for (cnt = 0; cnt < 10; cnt++) {
    LED = ON;
    Delay_ms(50);
    LED = OFF;
    Delay_ms(50);
}
Lcd_Custom_Out(1, 1, "          ");
}

//*****
*

void main()
{
    short          cnt;
    //ポートの設定
    TRISA = 0b11111111;
    TRISB = 0b00000000;
    TRISC = 0b10010000;
    TRISD = 0b11111111;
    ADCON1.PCFG3 = 0;
    ADCON1.PCFG2 = 1;
    ADCON1.PCFG1 = 1;
    ADCON1.PCFG0 = 1;
    CMCON.CM2 = 1;
    CMCON.CM1 = 1;
    CMCON.CM0 = 1;
    //□□□の初期化
    Lcd_Custom_Config(&PORTB,7,6,5,4,&PORTB,1,2,3);
    Lcd_Custom_Cmd(LCD_CURSOR_OFF);
    Lcd_Custom_Cmd(LCD_CLEAR);
    for (cnt = 1; cnt <= 16; cnt++) {
        Lcd_Custom_Chrc(1, cnt, 0xFF);
        LED = ON;
        Delay_ms(30);
        LED = OFF;
        Delay_ms(30);
    }
    for (cnt = 1; cnt <= 16; cnt++) {
        Lcd_Custom_Chrc(2, cnt, 0xFF);
        LED = ON;
        Delay_ms(30);
    }
}
```

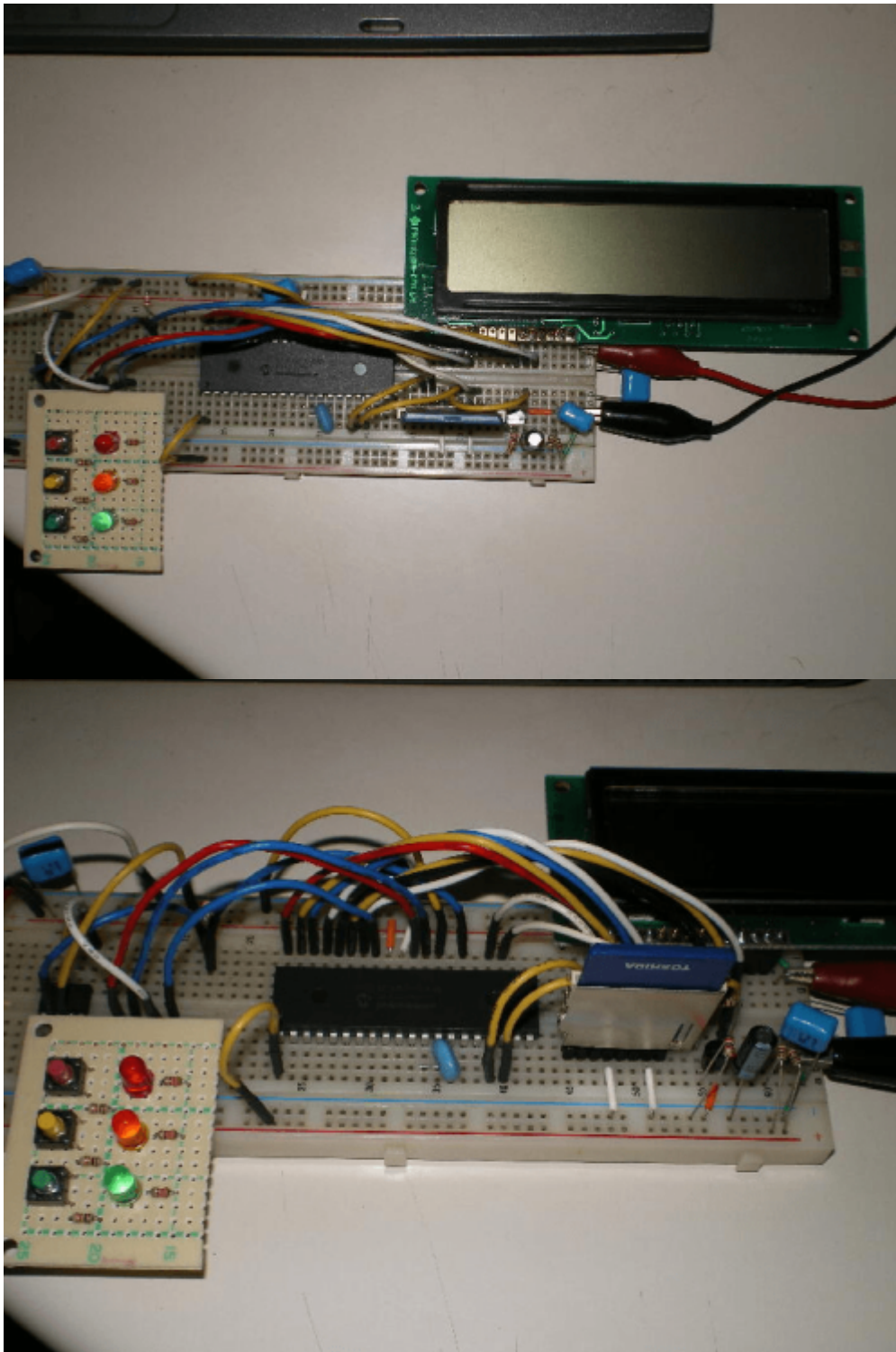
```

        LED = OFF;
        Delay_ms(30);
    }
    Lcd_Custom_Cmd(LCD_CLEAR);
    //□□□の初期化
    Spi_Init_Advanced(MASTER_OSC_DIV64, DATA_SAMPLE_MIDDLE,
CLK_IDLE_LOW, LOW_2_HIGH);
    if (Mmc_Fat_Init(&PORTC, 2)) {
        Lcd_Custom_Out(1, 1, "MMC error!");
        while (1) {
            LED = ON;
            Delay_ms(50);
            LED = OFF;
            Delay_ms(50);
        }
    }
    Spi_Init_Advanced(MASTER_OSC_DIV16, DATA_SAMPLE_MIDDLE,
CLK_IDLE_LOW, LOW_2_HIGH);
    //□□□□□□の初期化
    Usart_Init(9600);
    Delay_ms(100);
    //割り込みの設定
    PIE1.RCIE = 0;
    PIR1.RCIF = 0;
    INTCON.PEIE = 0;
    INTCON.GIE = 0;
    //
    while (1) {
        //開始スイッチが押されるのをチェックする。
        while (SW_START == 1) {
            Delay_ms(10);
        }
        //
        if (SW_MODE == 1)
            SDCwriter();
        else
            SDCreader();
    }
} //~!

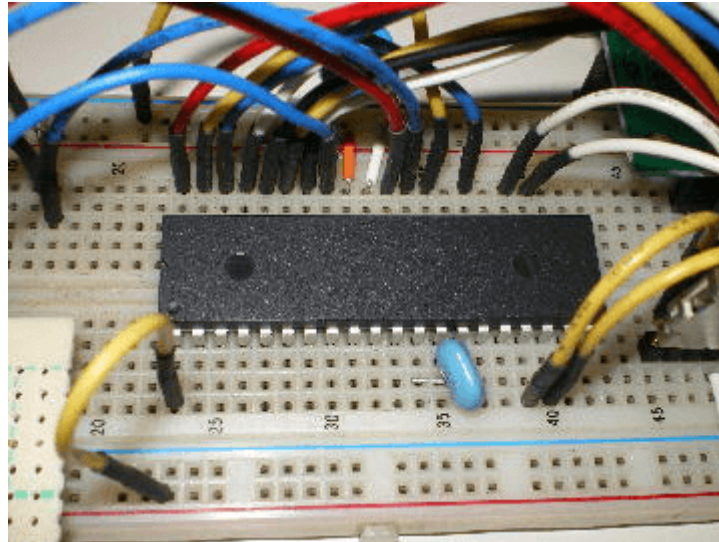
//*****
*
```

動作確認

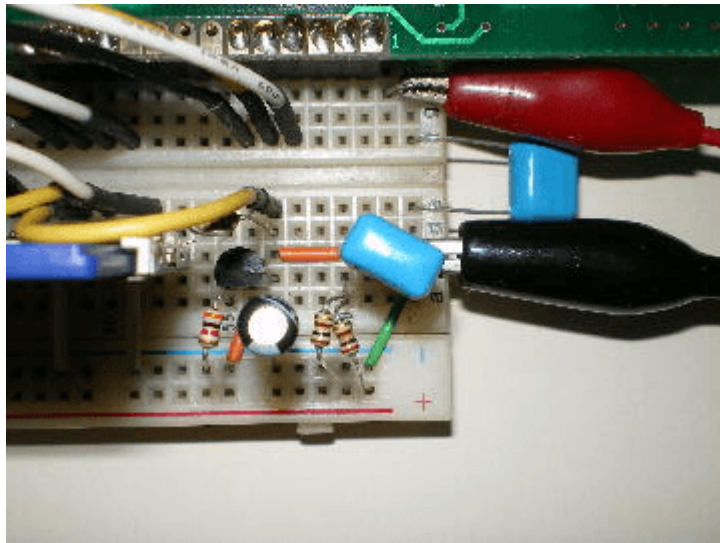
いつものブレッドボードで動作を確認しました。左側の基板は、製作上よく使う、プッシュスイッチ(3個)□LED(3個)をユニット化(治具)したものです。



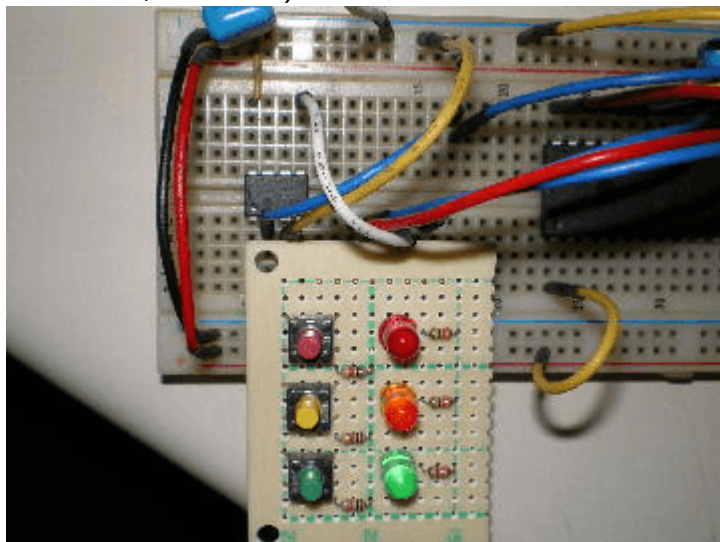
左側:PIC18F4520周辺です。右側:SDカード用の電源となる3.3Vを、5Vから変換する、シャント・レギュ

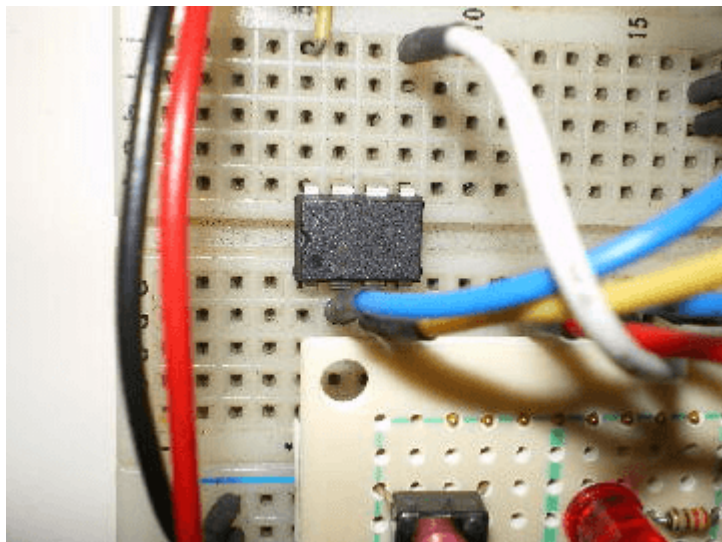


レータTL431周辺です。



RS232Cのテストデータを発生させるPIC12F683です。これによって、80文字のデータ+CR/LFコードをRS232C(TTLレベル)として、連続的に本装置へ送信します。

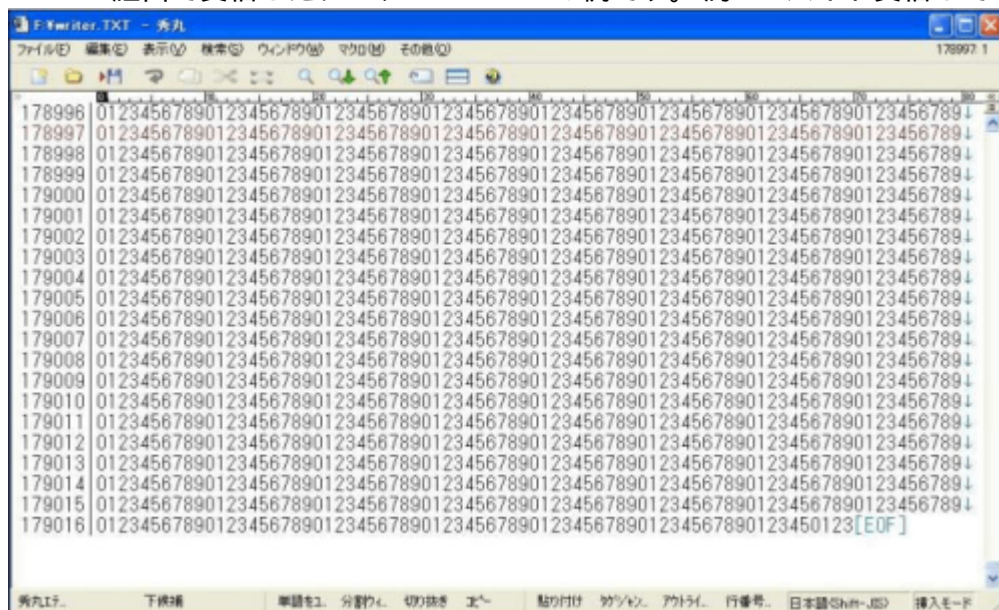




SDカード上のファイルです。reader.txtは、本装置が、RS232C経由でデータを送信するためのものです。writer.txtは、本装置が、RS232C経由で受信したデータを保存するためのものです。



RS232C経由で受信したデータ“writer.txt”の例です。約15Mバイト受信してみました。約4時間分¹⁾です。



参考

RS232Cのテストデータを発生させるPIC12F683のソフトです。単純に、80文字のデータ+CR/LFコードを送信するだけのものです。

```

1 //*****
2 /*↓
3  □□『RS232Cテストデータ送信用』↓
4 */↓
5 //*****
6 ↓
7 #define SW_START GPIO.F0↓
8 #define SW_STOP GPIO.F1↓
9 #define SW_SPEED GPIO.F4↓
10 ↓
11 #define LED GPIO.F2↓
12 ↓
13 #define CR 0x0D↓
14 #define LF 0x0A↓
15 ↓
16 #define ON 1↓
17 #define OFF 0↓
18 ↓
19 //*****
20 ↓
21 void main()↓
22 {↓
23   char cnt1, cnt2;↓
24   //↓
25   OSCCON = 0b01110000; // クロックは8Mhz ↓
26   ANSEL = 0b00000000; // 今回は使用しない。↓
27   CMCON0 = 0b00000111; // 今回は使用しない。↓
28   TRISIO = 0b00011011;↓
29   OPTION_REG.F7 = 0; // PORTをプルアップ設定する。 ↓
30   WPU.F0 = 1;↓
31   WPU.F1 = 1;↓
32   WPU.F4 = 1;↓
33   Soft_Uart_Init(GPIO, 3, 5, 9600, 0);↓
34   LED = OFF;↓
35   while (1) {↓
36     while (SW_START == 1) {↓
37       Delay_ms(10);↓
38     }↓
39     //↓
40     while (SW_STOP == 1) {↓
41       LED = ON;↓
42       for (cnt1 = 0; cnt1 < 8; cnt1++) {↓
43         for (cnt2 = 0; cnt2 < 10; cnt2++) {↓
44           Soft_Uart_Write(cnt2 + '0');↓
45         }↓
46       }↓
47       Soft_Uart_Write(CR);↓
48       Soft_Uart_Write(LF);↓
49       LED = OFF;↓
50       //↓
51       if (SW_SPEED == 0) {↓
52         Delay_ms(100);↓
53       }↓
54     }↓
55   }↓
56 }↓
57 ↓
58 //*****
59 [EOF]

```

1)

180000バイト × 82バイト ÷ (9600bps ÷ 10

Last update:
2025/10/17 14:27

elechobby:picdic:otherpic:183 <http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:otherpic:183&rev=1588245853>

From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:otherpic:183&rev=1588245853>

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

