

簡易レベルメータ(8チャンネル×8ドット)

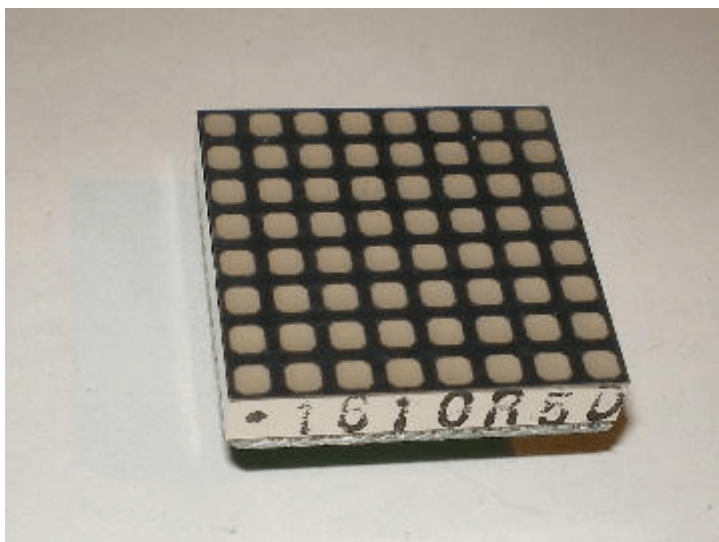
概要

ドットマトリクスタイプのLEDが、安価(100円~200円程度)で手に入るようになりました。何個か並べれば、表示装置として使えそうですが、今回は、1個だけ使用して、簡単なレベルメータを製作してみました。

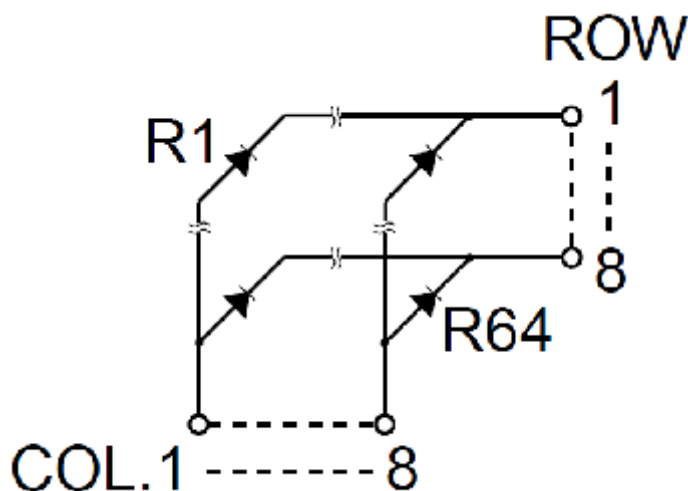
使用したLEDドットマトリクス“BU5004-R”は、64ドット(8×8)の赤色LEDタイプのものです。他にも同様の物があり使用できます。

<仕様>

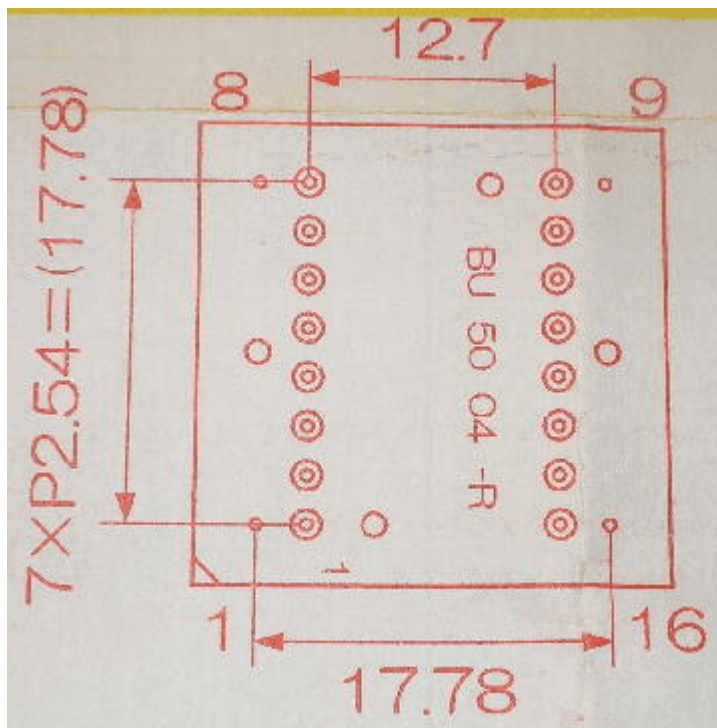
- 8チャンネルのアナログデータを入力し、BU5004-Rにバー表示します。
- バー表示は、チャンネルあたり、8ドットです。
- 起動時に、オープニングデモを表示します。



<BU5004-Rの概観>



<BU5004-RのCOL(列)とROW(行)>



<BU5004-Rのピン配置(1)>

ROW	1	2	3	4	5	6	7	8
PIN No.	16	1	14	13	5	6	9	8
COL	1	2	3	4	5	6	7	8
PIN No.	7	4	3	2	10	11	12	15

<BU5004-Rのピン配置(2)>

動作原理

<ダイナミック点灯処理>

- TIMER0を使用して、約1msecの割り込みを発生させます。
- 割り込み処理の中で、行毎(ROW1~ROW8)[]列毎(COL1~COL2)にLEDを点灯させます。

<レベルメータ処理>

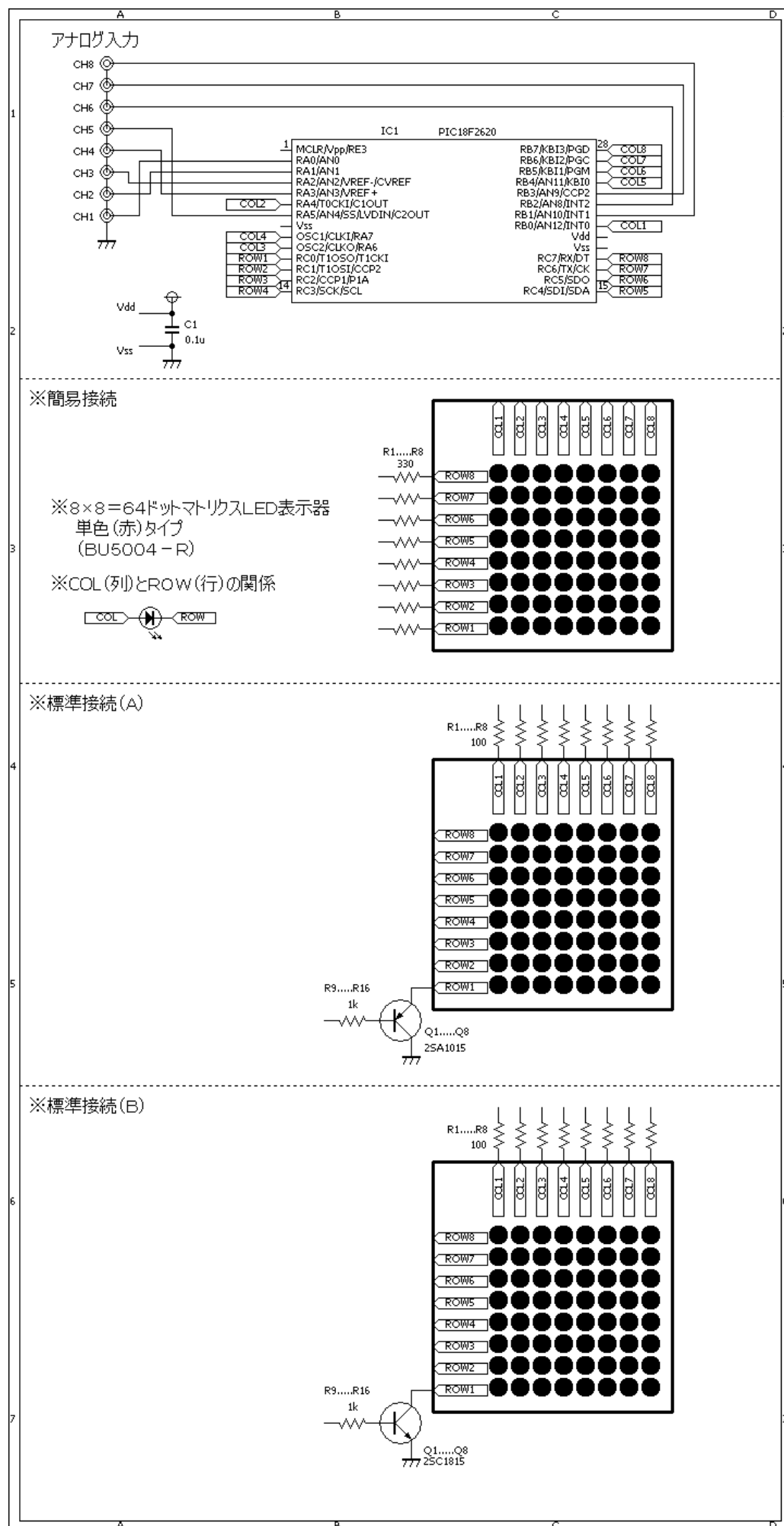
- アナログデータを取り込み、バー表示します。
- バー表示は、チャンネルあたり8ドットを使用し、9段階(全消灯~全点灯まで)とします。
- 従って、分解能は約557mVになります。
※ $557\text{mV} = (5000\text{mV} \div 1024) \times (1024 \div 9)$

<オープニングデモ処理>

- 全点灯、全消灯を10回繰り返します。
- ランダムに点灯消灯を1000回繰り返します。
- 行順に点灯させます。
- 行順に消灯させます。
- 列順に点灯させます。
- 列順に消灯させます。

回路図

BU5004-Rの接続は、“簡易接続方式”としています。PICの各I/Oピンの最大電流は、25mAなので、これを超えない範囲で電流制限抵抗(R1~R8)を調整してください。(200Ω~1kΩ) “標準接続方式(A)”の場合には、制御論理が同じなので、プログラムの修正は不要です。 “標準接続方式(B)”の場合には、制御論理が逆なので、プログラムを少し修正する必要があります。



ソースコード

[led8x8.c](#)

```
//*****
*
/*
    <簡易レベルメータ ( 8 チャンネル× 8 ドット ) >
*/
//*****
*
extern void    main();
extern void    interrupt();
extern void    opening_demonstration();
//*****
*
char    bar_col[9] = {
    0b00000000,
    0b10000000,
    0b11000000,
    0b11100000,
    0b11110000,
    0b11111000,
    0b11111100,
    0b11111110,
    0b11111111
};
char    bar_row[9] = {
    0b11111110,
    0b11111101,
    0b11111011,
    0b11110111,
    0b11101111,
    0b11011111,
    0b10111111,
    0b01111111,
    0b11111111
};
char    bar_data[8];
//*****
*
//■■■メイン関数■■■
void    main()
{
    //クロックを8MHz に設定します。
    OSCCON.IRCF2 = 1;
    OSCCON.IRCF1 = 1;
    OSCCON.IRCF0 = 1;
    //
```

```

TRISA = 0b00101111;
TRISB = 0b00001110;
TRISC = 0b00000000;
//□□□変換を使用します。
ADCON1.PCFG3 = 0;
ADCON1.PCFG2 = 1;
ADCON1.PCFG1 = 0;
ADCON1.PCFG0 = 0;
//□□□□□を設定します。
T0CON.T0CS = 0;
T0CON.PSA = 0;
T0CON.T0PS2 = 0;
T0CON.T0PS1 = 1;
T0CON.T0PS0 = 0;
T0CON.TMR0ON = 1;
INTCON.TMR0IE = 1;
INTCON.TMR0IF = 0;
// 割り込みを許可します。
INTCON.PEIE = 1;
INTCON.GIE = 1;
//
opening_demonstration();
//
while (1) {
    bar_data[0] = bar_col[Adc_read(0) / 114];
    bar_data[1] = bar_col[Adc_read(1) / 114];
    bar_data[2] = bar_col[Adc_read(2) / 114];
    bar_data[3] = bar_col[Adc_read(3) / 114];
    bar_data[4] = bar_col[Adc_read(4) / 114];
    bar_data[5] = bar_col[Adc_read(8) / 114];
    bar_data[6] = bar_col[Adc_read(9) / 114];
    bar_data[7] = bar_col[Adc_read(10) / 114];
    Delay_ms(50);
}
}
//*****
*
//■■■割り込み関数■■■
short bar = 0;
void interrupt()
{
    if (INTCON.TMR0IF == 1) {
        INTCON.TMR0IF = 0;
        //
        switch (bar) {
            case 0:
            case 1:
            case 2:
            case 3:
            case 4:
            case 5:

```

```
        case 6:
            PORTC = bar_row[8];
            PORTB = bar_data[bar];
            PORTA.F4 = bar_data[bar].F1;
            PORTA.F6 = bar_data[bar].F2;
            PORTA.F7 = bar_data[bar].F3;
            PORTC = bar_row[bar];
            bar++;
            break;

        case 7:
            PORTC = bar_row[8];
            PORTB = bar_data[bar];
            PORTA.F4 = bar_data[bar].F1;
            PORTA.F6 = bar_data[bar].F2;
            PORTA.F7 = bar_data[bar].F3;
            PORTC = bar_row[bar];
            bar = 0;
            break;
    }
}

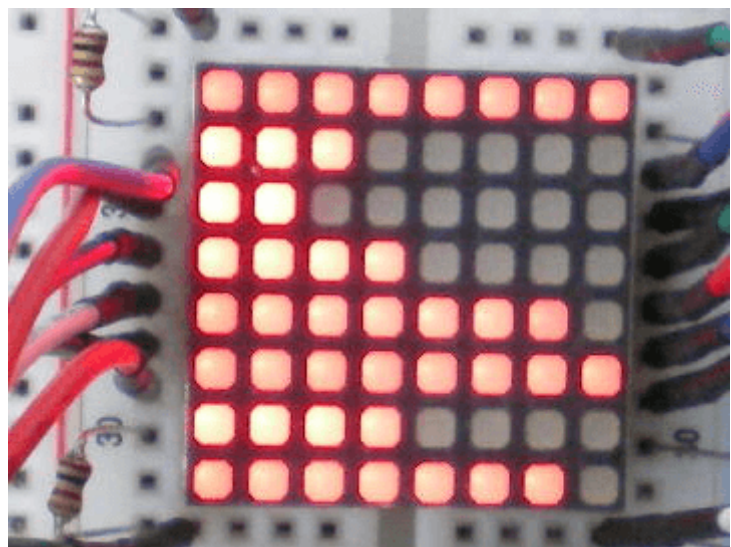
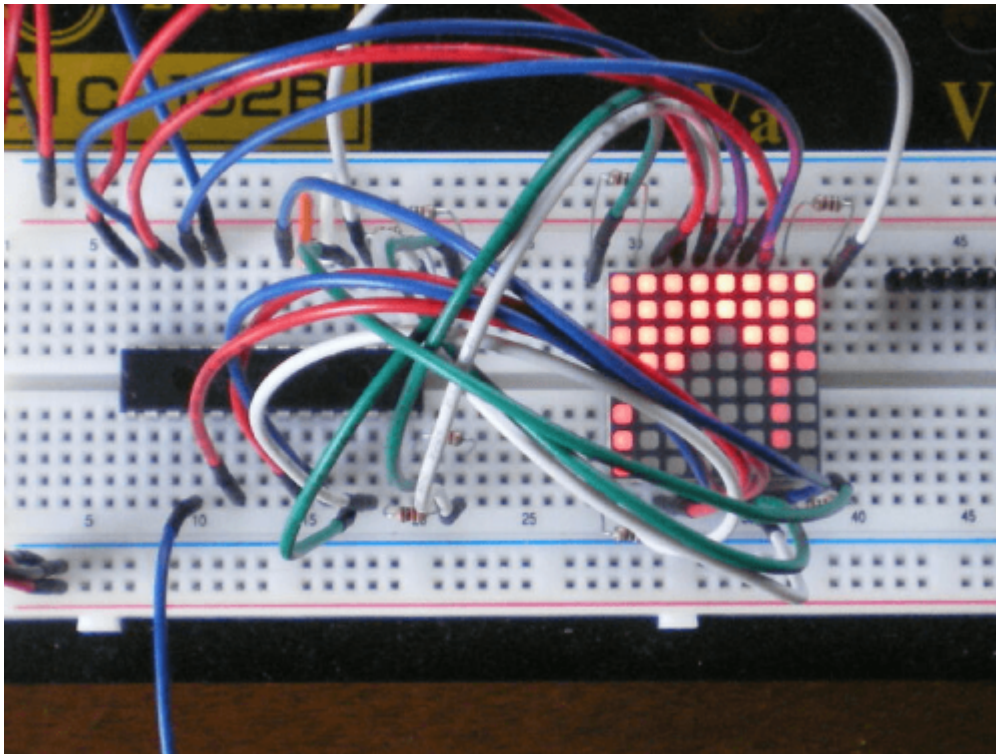
//*****
*
//■■■オープニングデモ関数■■■
void opening_demonstration()
{
    short cnt1, cnt2;
    int cnt;
    //
    for (cnt2 = 0; cnt2 < 10; cnt2++) {
        for (cnt1 = 0; cnt1 < 8; cnt1++) {
            bar_data[cnt1] = bar_col[8];
        }
        //
        Delay_ms(100);
        //
        for (cnt1 = 0; cnt1 < 8; cnt1++) {
            bar_data[cnt1] = bar_col[0];
        }
        //
        Delay_ms(100);
    }
    //
    for (cnt = 0; cnt < 1000; cnt++) {
        bar_data[rand() / 4096] = rand() / 128;
        Delay_ms(10);
    }
    //
    for (cnt1 = 0; cnt1 < 8; cnt1++) {
        for (cnt2 = 0; cnt2 < 9; cnt2++) {
            bar_data[cnt1] = bar_col[cnt2];
        }
    }
}
```

```
        Delay_ms(10);
    }

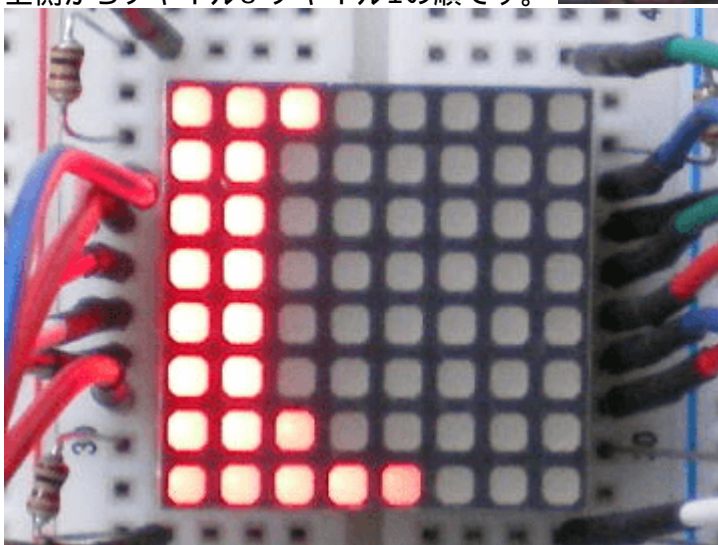
}
//
for (cnt1 = 0; cnt1 < 8; cnt1++) {
    for (cnt2 = 8; cnt2 >= 0; cnt2--) {
        bar_data[cnt1] = bar_col[cnt2];
        Delay_ms(10);
    }
}
//
for (cnt2 = 0; cnt2 < 9; cnt2++) {
    for (cnt1 = 0; cnt1 < 8; cnt1++) {
        bar_data[cnt1] = bar_col[cnt2];
        Delay_ms(10);
    }
}
//
for (cnt2 = 8; cnt2 >= 0; cnt2--) {
    for (cnt1 = 0; cnt1 < 8; cnt1++) {
        bar_data[cnt1] = bar_col[cnt2];
        Delay_ms(10);
    }
}

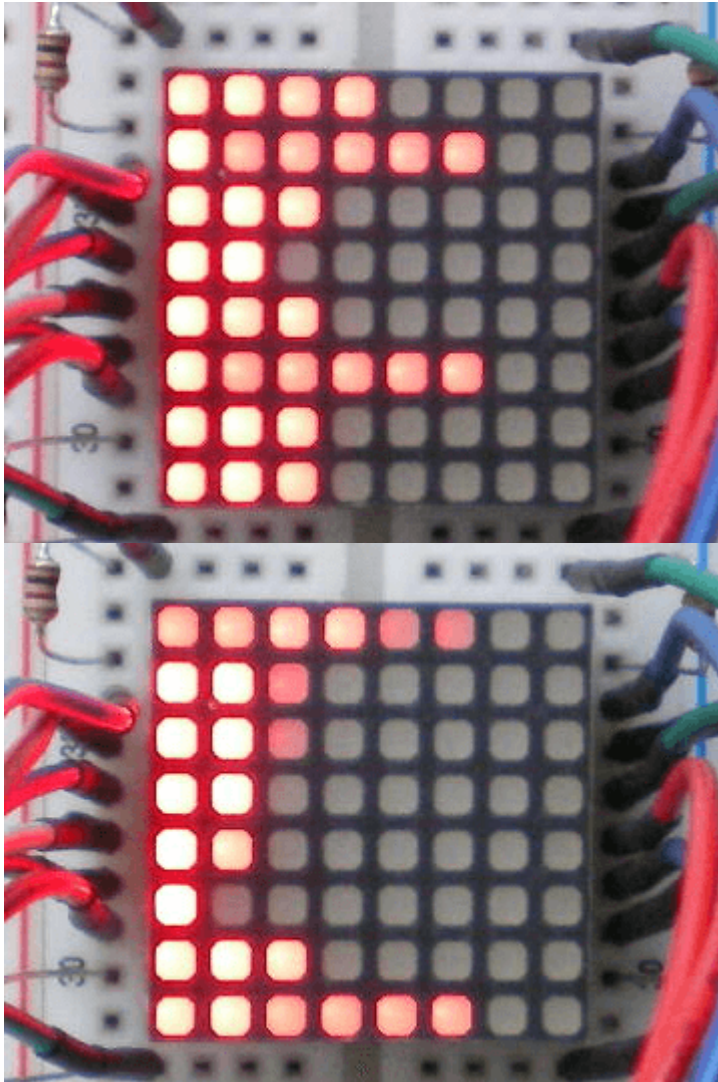
}
//*****
*
```

動作確認



上側からチャンネル8~チャンネル1の順です。





From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:otherpic:188&rev=1588249592>

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

