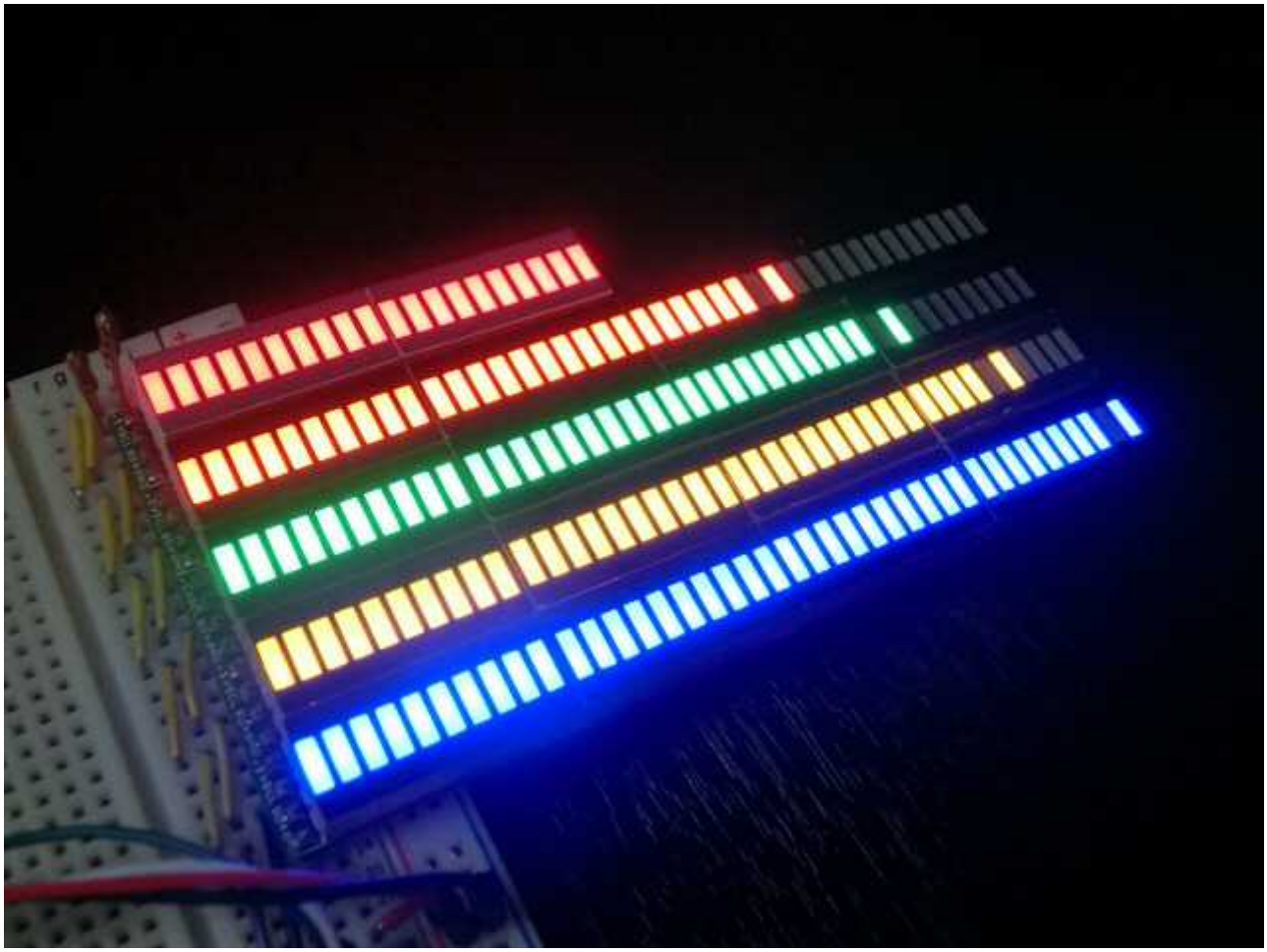
高速I2C型 20/30/40連LEDアレイモジュール

10連LEDアレイを2個～4個実装し、20 / 30 / 40連表示を高速I2C通信で簡単に制御できるモジュールです。
※ハンダゴテによる半田付け作業が必要になります。



1. 注意

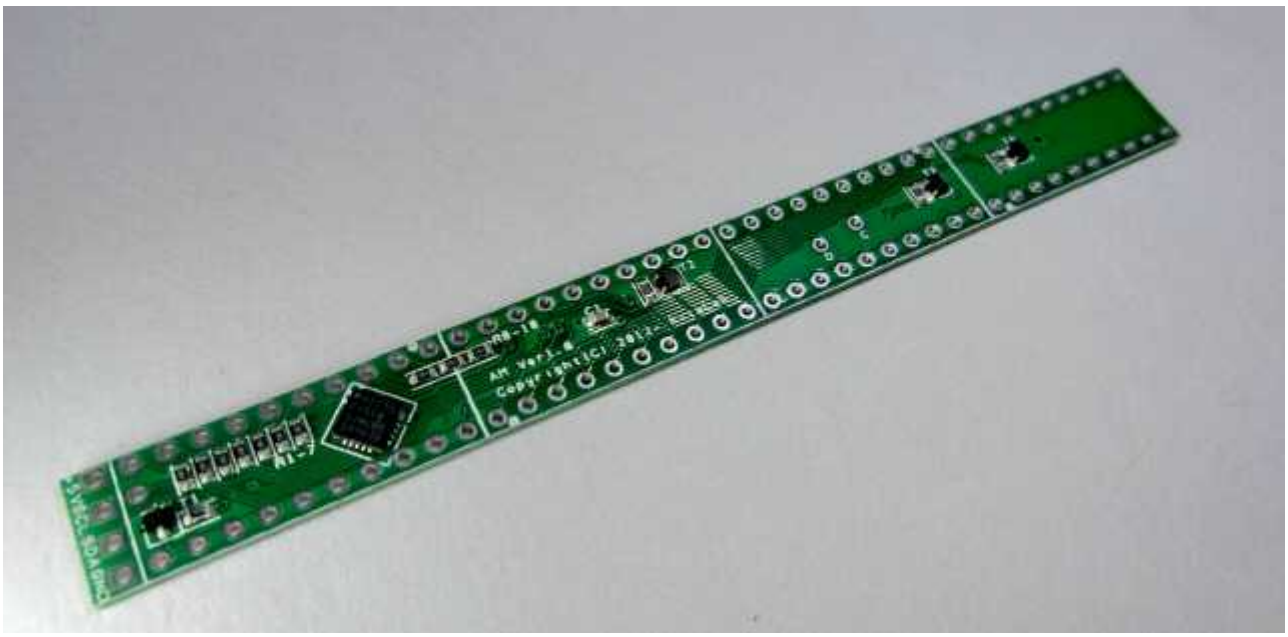
本品は、ハンダゴテによるハンダ付け作業が必要になります。
またその際、LED アレイの実装方法が若干特殊なので、必ず以降の説明をご覧下さい。

本品には、10 連 LED アレイ、及びピンヘッダは付属しません。別途ご用意下さい。
（付属する商品もございます）

2. 特徴

① 外部接続端子以外は 10 連 LED アレイ内にスッキリ回路が収まるようスレスレまでコンパクト設計されています。
② モジュール基板裏面の白線部分で切断することにより、20 連、30 連、40 連のいずれか好きな実装ができます。
③ 高速ダイナミック点灯により、極めてちらつきの少ない表示ができます。
④ 8bit×5個のビット転送機能により、40 個の LED を個別に ON/OFF 制御できます。
⑤ 非ビット転送機能では、バー表示、1点表示、ピーク表示、移動速度設定など色々な機能が搭載されています。
⑥ ピーク表示や自動減衰表示機能にも対応しているので、VU メータなどの面倒な視覚効果も数値を出力するだけです。
⑦ バーの移動速度、ピークバーの移動速度、ピークの保持時間なども自由に設定できます。
⑧ 左右反転機能が付いているので、どの向きでもご利用できます。
⑨ 7bit / 400Kbps の高速 I2C シリアル通信に対応しています。(実際にはもっと高速でも耐えられます)
⑩ 電源線 (+5v、GND) と I2C 通信線 (SCL、SDA) のたった4本だけでモジュールを動かせます。
⑪ I2C の最大の特徴である、複数のモジュールを並列に接続し、個別に制御できます。
⑫ 当モジュールの I2C スレーブアドレスは標準で「0xD0」ですが、マルチアドレスに対応していますので、0x10～0xEE まで好みの偶数スレーブアドレスを割り当てることができます。(変更したスレーブアドレスはフラッシュメモリに記憶されますので、電源を切っても保持されます)
⑬ 接続端子は縦一列4個なので、ブレッドボードでも大変扱いやすいです。
⑭ LED アレイは実装されていませんので、好きな色の LED アレイを用意して実装できます。(フルカラー型の LED アレイは独立構造ではありませんので当モジュールでは使用できません。 ご注意下さい)

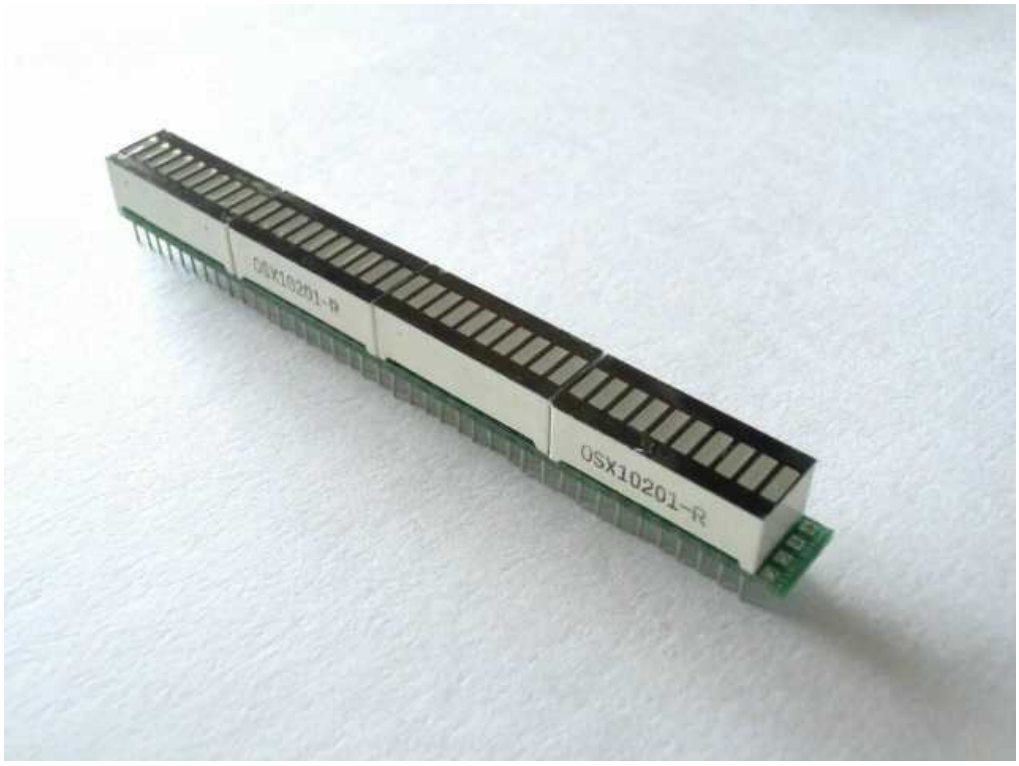
3. 内容

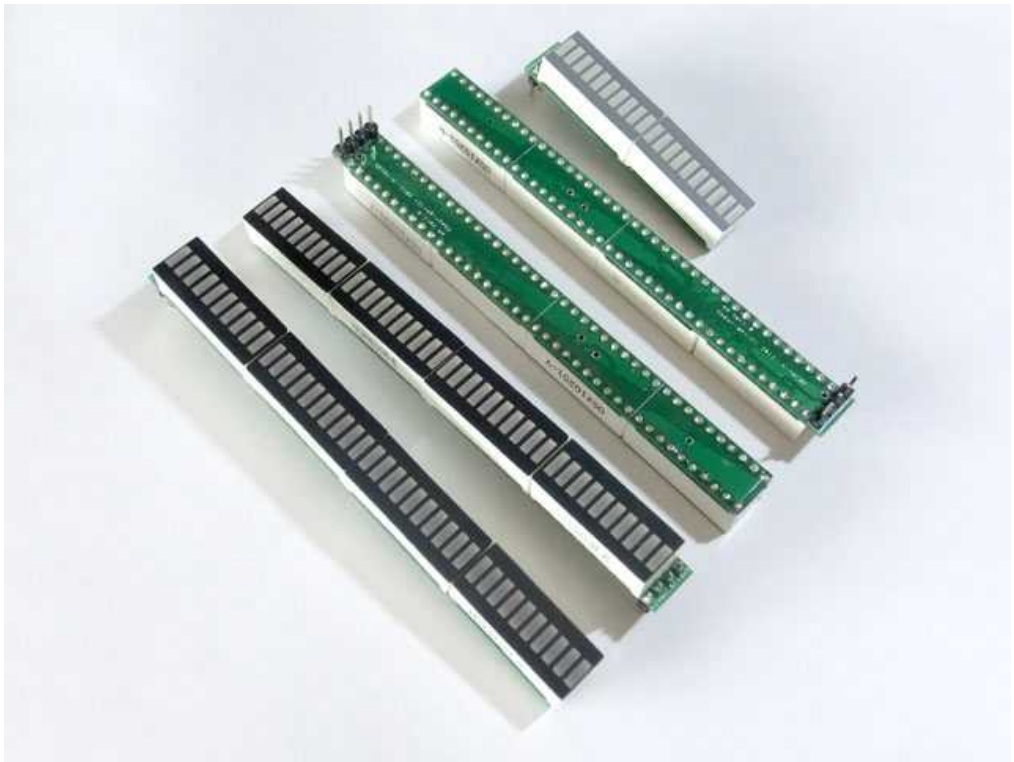


※ こちらが表面です。
※ 制御 IC、コンデンサなどのチップ部品は実装済みです。
※ 写真のものが全てです。 LED アレイ、及びピンヘッダは付属しませんので、別途ご用意下さい。
(付属する製品も一部ございます)

10 連 LED アレイを2個～4個用意し、ハンダ付けするだけでご利用できます。
※ LED アレイを実装する向きが特殊なので、必ず以降の説明をご覧になりながら作業をして下さい。

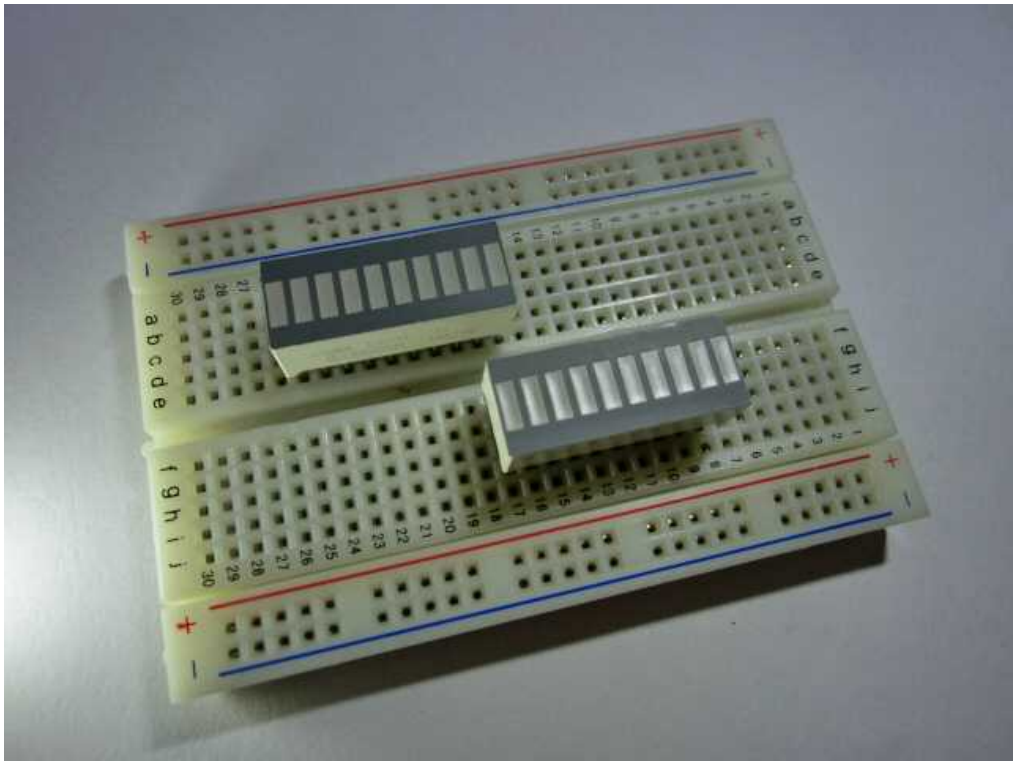
4. 完成例





※ モジュール基板裏面の白線に沿ってプリント基板を切断することで、20 連、30 連表示モジュールにすることもできます。但し、切断面がショートすると表示不良の原因になりますので、切断にはテーブルソー＋ダイヤモンド歯などのご利用をお薦めします。 また、切断面にはソルダーレジスト等を塗布することをお薦めします。

5. 実装可能な 10 連 LED アレイ



「10 連 LED アレイ」、または「10 バーLED アレイ」と呼ばれている LED アレイで、10 個全てが独立した配線になっていることが前提です。 フルカラー表示に対応した LED アレイは配線が独立しておらず当モジュールでは使用できませんのでご注意下さい。

LED の制限抵抗として、1LED あたり 180Ω の抵抗器がモジュール上に実装されています。また LED に流れる電圧は入力電圧になります。

LED アレイによっては、横置きにした場合に左右が非対称、または上下非対称の製品があります。上下非対称の場合は実装時に位置が合うよう補正し、左右非対称の場合は LED アレイが当モジュールに若干入れにくく、また実装してみると当モジュールが軽く反りますので判ります。その場合は、LED アレイの厚みのある面をヤスリ等で軽く削って調整をしてから実装して下さい。

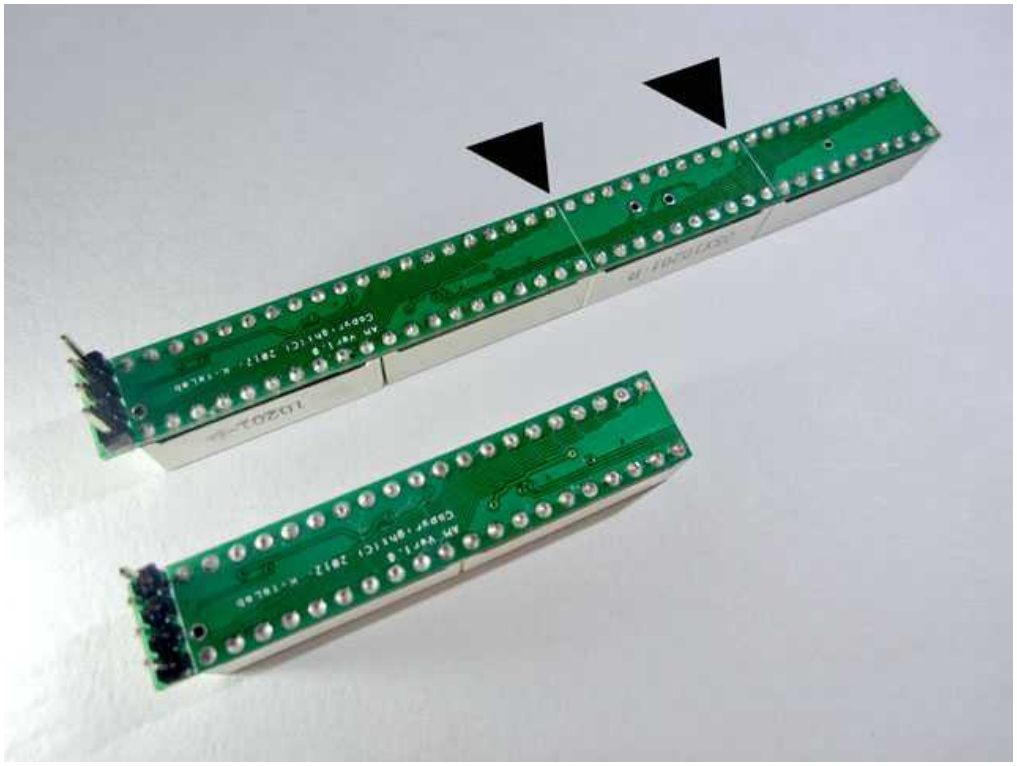
6. 当モジュールの使用方法

当モジュール基板の端に、+5v、SCL、SDA、GND の 4 つの端子が存在します。 通信および電源として必要なのはこの 4 端子だけです。 +5v には DC5V（±10%）の安定電源を※。 SCL、SDA は I2C バスに接続して下さい。（I2C バスはプルアップされている必要があります）

※ 当モジュールの許容電圧は、3V～5V ですので 3V でも動作はしますが、保証外です。
※ 出荷状態では、+5V と GND に電源を接続するだけで、初期点灯試験が実行されます。

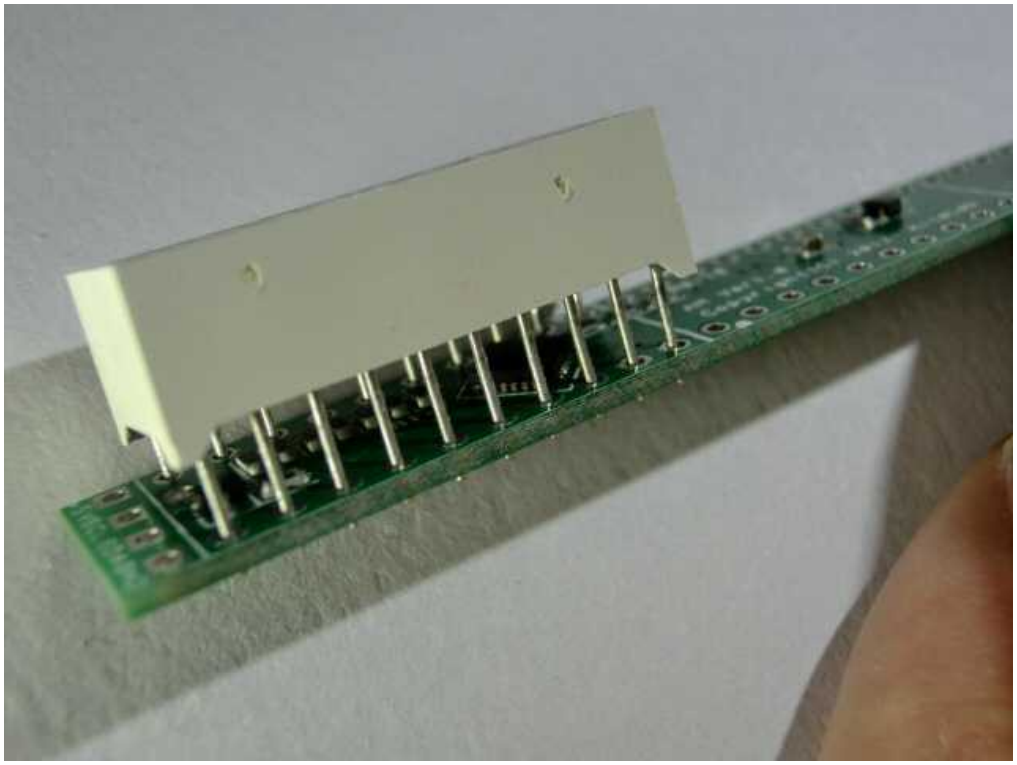
7. LED アレイの実装手順

LED アレイの実装には、いくつか注意点があります。
まずは、何個実装するか。

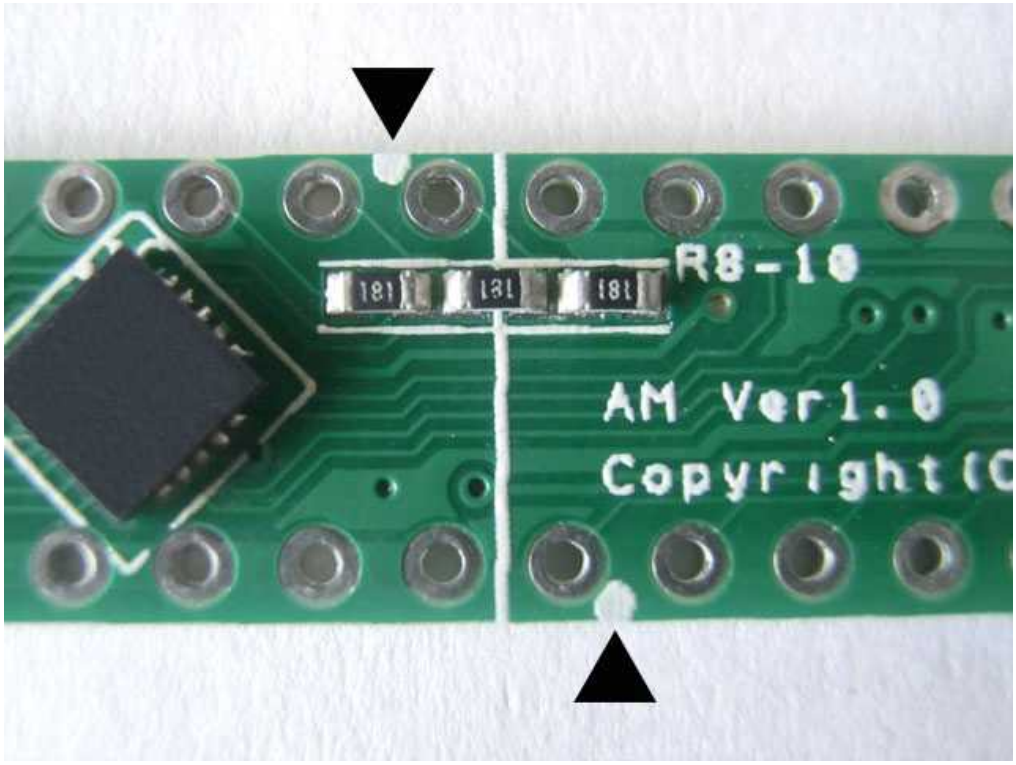


当モジュールは、基板裏面の白線（上の写真の▼の箇所）に沿って切断することで、20 連モジュール化、または 30 連モジュール化することができます。
尚、前述（4章）の通り、切断面がショートすると表示不良の原因になりますので、切断にはテーブルソー＋ダイヤモンド歯などのご利用をお薦めします。 また、切断面には劣化防止のソルダーレジスト等を塗布することをお薦めします。

次に、LED アレイを実装する基板の面ですが、制御 IC などのチップパーツが実装されている表面に付けます。全てのチップパーツが LED アレイの裏の隙間に干渉せず収まるよう設計されていますので、しっかり奥まで挿して下さい。

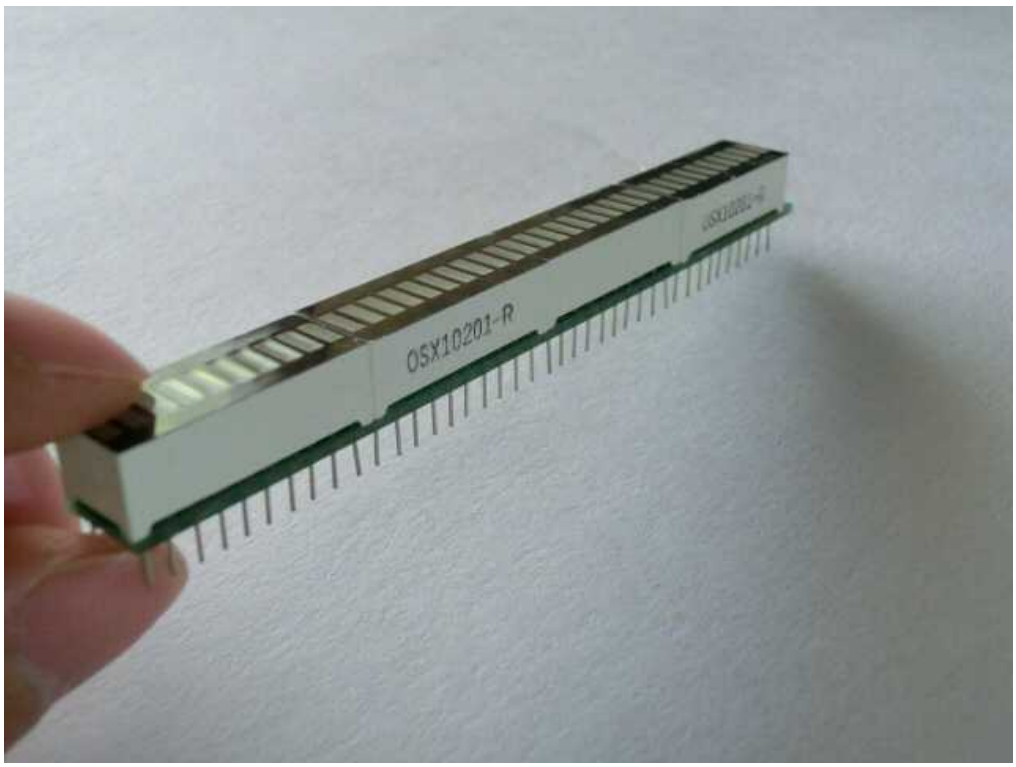


次に、LED アレイの向きですが、LED アレイの1番が、モジュール基板表面に描かれた白点にくるように実装します。厳密には、白点のある方がアノードになります。下の写真の▼の部分にあるのが白点です。

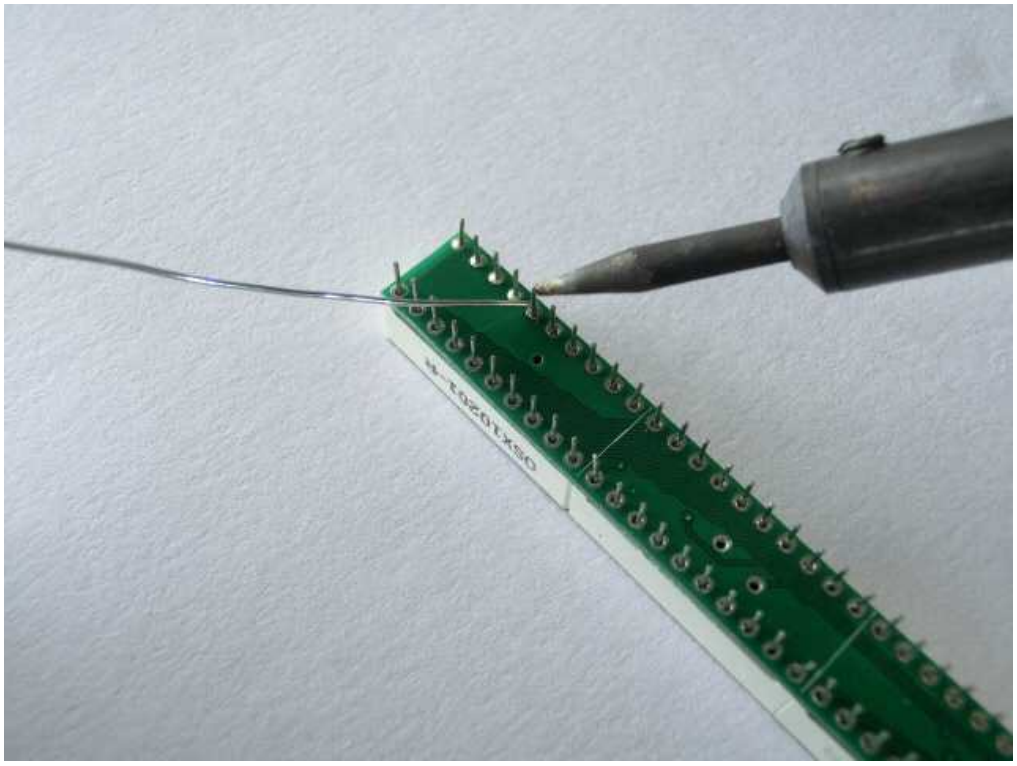


上の写真をご覧いただくと判る通り、LED アレイ毎に白点（1番ピン）の向きが上下逆になっています。 これはコンパクト化の都合ですので、間違ってすべて同じ向きにしないよう、十分にご注意下さい！

図で表すと、下の写真のように、1個毎に向きが逆になります。
（文字の書かれた面がアノードとは限りませんので、十分に確認をした上で実装して下さい。可能であれば LED アレイをハンダ付けする前に、一度+5V と GND に通電し、初期点灯試験で点灯するか確認してみることをお薦めします）

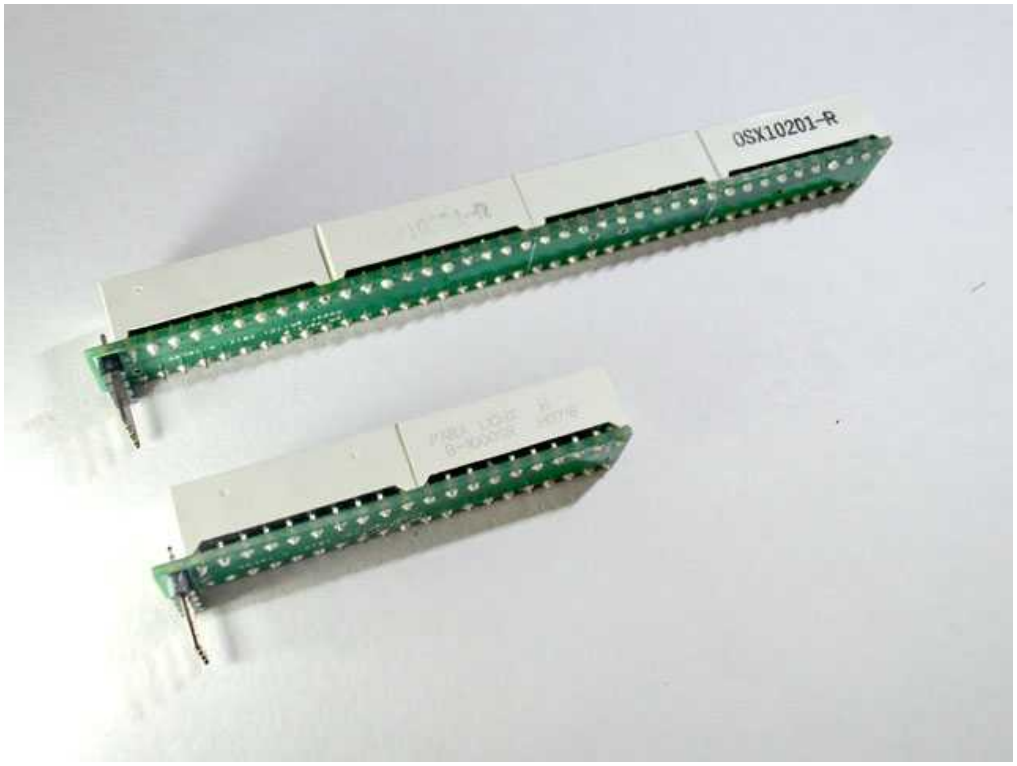


LED アレイの実装向きを確認し、綺麗に1列になるよう位置を微調整したら、あとはひたすらハンダ付けです。



最後に4端子のピンヘッダをハンダ付けしたら完成です。
ピンヘッダは、ブレッドボードなどに挿しながら作業するとハンダ付けし易いです。

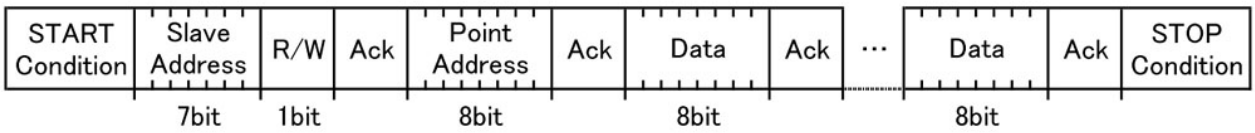
尚、ユニバーサル基板等に実装する場合は、LED アレイの長い足を、当モジュールの固定用のピンとして流用頂くのがベストですが、ブレッドボードでの運用がメインの場合は、LED アレイの足を切断した方が使い易いです。



LED アレイの足を切断する際は、当モジュールのプリント基板に傷を付けないよう、十分にご注意下さい。
切断後の裏面には、絶縁兼クッション目的の保護マットを貼ると良いと思います。

8. 当モジュールの I2C 通信の流れ

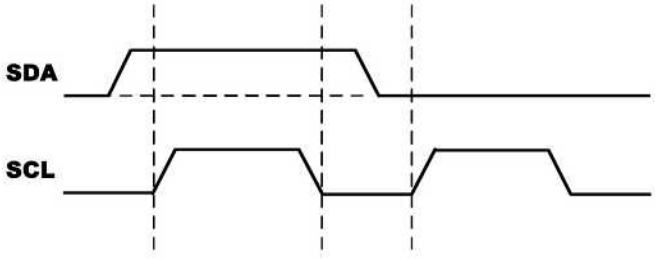
① START Condition を送信。
② 7bit の I2C スレーブアドレスを送信。
③ 1bit の Read/Write 命令を送信。(Write は 0 なので 0 を送信)
④ Ack 処理。
⑤ 8bit の機能アドレスを送信し、Ack 処理。
⑥ 8bit のデータを送信し、Ack 処理。 (以降、必要なだけ⑥を繰り返し)
⑦ STOP Condition を送信。



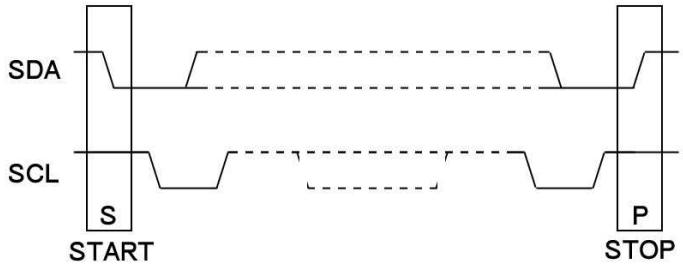
R/W は Read の場合は 1 を、Write の場合は 0 を指定し、7bit の Slave Address と組み合わせて扱います。

I2C はデータの同期受信確認用にスレーブ (当モジュール) からアクノリッヂ (Ack) 情報が返されます。アクノリッヂは Lo アクティブですので、マスター側は SDA を Hi にして待ちます。送出した 8bit の情報をレシーブが正しく受信すると SDA はスレーブにより Lo にされますので次のデータの送信を始めます。

I2C のデータ送出 (1bit 送出) の基本は、SDA (データライン) を Hi(1)または Lo(0)にしてから SCL (クロックライン) を Hi にして、再び SCL を Lo にしてから次のビットを SDA に指示します。 簡単なプロトコルなので、マイコンがハードウェア的に I2C マスターに対応していなくても、ソフトウェアだけで容易に実装できる点が特徴です。



START Condition、及び STOP Condition は、I2C 通信の開始と終わりを示すものです。 データ送出の場合は SCL が Lo の時に SDA を変更しましたが、START / STOP Condition の場合は、SCL が Hi の状態で SDA を Hi→Lo にすると START Condition、逆に Lo→Hi にすると STOP Condition になります。



※I2C は同期通信ですが、レシーブの性能を超える速度で通信をしようとすると、当然正しく通信はできませんので、各々要所に若干のウェイト入れが必要です。

9. 当モジュールの I2C 機能アドレステーブル

Point Address		Description
	0x00 (※1)	点灯位置設定。(0～40)　このアドレスに送信した数値の位置、または数値分の LED アレイを点灯させます。
	0x02 (def=0x00)	モード設定。 8bit のビットフラグで機能を選択します。 Bit 7 : 自動減衰機能 0=OFF / 1=ON VU メータのように自動的に減衰します。減衰速度はバーの移動時間設定 (0x03) に依存し、点灯位置設定時 (0x00) は瞬時に点灯します。 Bit 6 : 1点表示機能 0=OFF / 1=ON 0 の時はバーグラフ表示、1 の時はポイント表示になります。 Bit 5 : 点灯反転機能 0=OFF / 1=ON 点灯 / 消灯状態が逆になります。 Bit 4 : 左右反転機能 0=OFF / 1=ON 0 の時は端子側が基準点、1 の時は無端子側が基準点になります。 Bit 3 - 0 : 未使用(常に 0)
	0x03 (def=0)	バー移動時間設定。(0～255)　バーが1ポイント移動するまでの時間を設定します。(設定値×約 200us がウェイト値となります)　※尚、このアドレスに「0」を設定した場合は移動機能が OFF になり、瞬時に点灯位置が変更されます。
	0x04 (def=0)	ピーク保持時間設定。(0～255)　ピークバーが最大値に留まり続ける時間を設定します。(設定値×約 200us がウェイト値となります)　※尚、このアドレスに「0」を設定した場合はピークバーの機能が OFF になります。
	0x05 (def=40)	ピークバー移動時間設定。(0～255)　ピークバーが1ポイント下がるまでの時間を設定します。(設定値×約 200us がウェイト値となります)
	0x06 (def=40)	上限設定。(0～40)　このアドレスに登録した数値以上の値をアドレス 0x00 で送信した場合は、丸め込みされます。
	0x10～14	ビットデータ設定。(40 連 = 8bit×5 バイト分) ※ このアドレスにデータを送信すると、数値関連の機能 (アドレス 0x00～0x06) がすべて OFF になり、送信したビット情報通りに点灯します。1 が点灯、0 が消灯で、モジュール上は、[接続端子：0x10 11 12 13 14]でマップされます。
	0xF0 (※2) (def=0xD0)	I2C スレーブアドレス変更指示。 (詳しくは10. I2C カスタムスレーブデバイスアドレスを参照)
	0xF2 (※2) (def=1)	起動時点灯試験モード設定。 (詳しくは11. 点灯試験モード設定を参照)
	0xF4 (※2) (def=4)	LED アレイ・ユニット数変更指示。 当モジュールを 20 連化、または 30 連化した際に左右反転機能を ON にすると、標準では 40 連位置を基準点として動作をしますので表示上不都合が生じます。その際は、ここで LED アレイを実装した数を2～4個の範囲で設定して下さい。

(def = デフォルト値)

※1 機能アドレスが連続していない機能は、1回毎に I2C ストップ、または再スタートの必要があります。

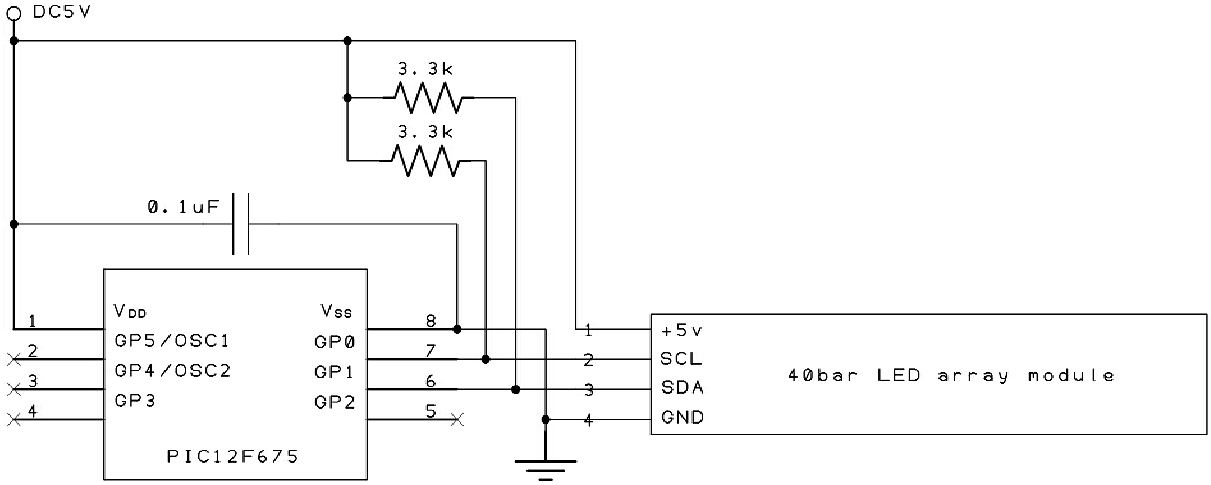
※2 0xF0、0xF2、0xF4 は、設定すると内部のフラッシュメモリに設定値が記憶されますので、電源を落としても設定したデータは保持されます。 また、設定毎にモジュールが再起動します。 尚、フラッシュメモリの書き換え回数には上限があり、メーカーの公称値では約10万回となっています。 いずれもそう頻繁に更新するような情報ではありませんが、10万回を超えるほど頻繁には更新するような処理はしないよう、ご注意下さい。

10. I2C カスタムスレーブデバイスアドレス設定 (PointAddress = 0xF0)

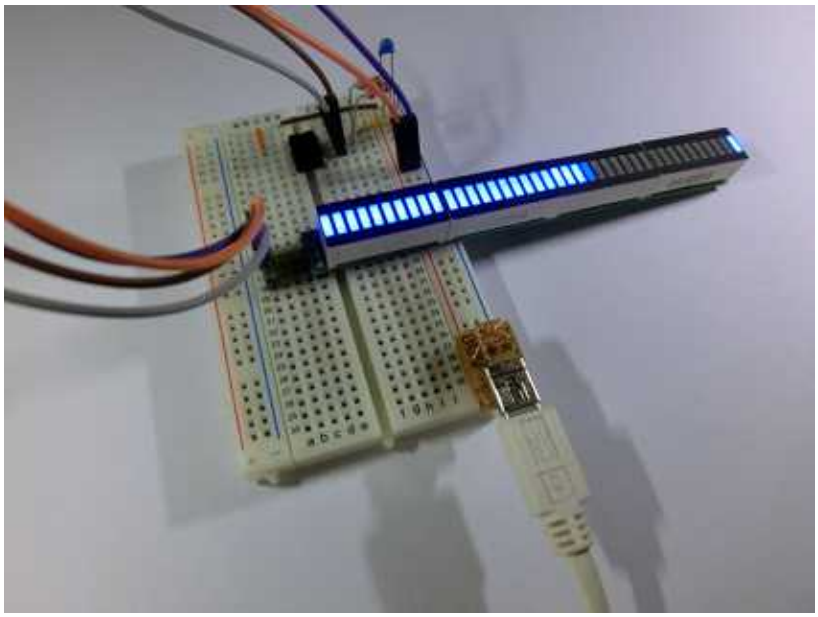
当モジュールの I2C スレーブデバイスアドレスは、標準では 0xD0 が割り当てられていますが、複数のモジュールを接続する場合や、他の I2C 機器とスレーブデバイスアドレスが競合していた場合に不都合が生じます。

そこで、当モジュールには、I2C のアドレスを自由に変更できる、カスタムアドレス設定機能が搭載されています。この機能を利用すれば 0x10～0xEE までの偶数アドレスが自由に設定できます。尚、当モジュールのスレーブアドレスを書き換えるためには、以下のような専用の小規模回路と簡単なプログラムを用意する必要があります。

[回路図]



ブレッドボードで組むと、この程度です。



[カスタムアドレス登録プログラム] - MPLAB HI-TECH C　main.c

```
// ===================================================================
// 40bar LED array module - customize program for PIC12F675
// ST:2012/07/28 ST:2012/07/28
// Copyright(C)2012- Kitalab / T.KITAKUBO
// http://store.kitalab.com/
// -------------------------------------------------------------------
// Compiler   : HI-TECH C Compiler for PIC10/12/16 MCUs (Lite Mode)  V9.83
// PIC        : PIC12F675
// Oscillator : INTRC/4MHz
// Power      : 5V
// ===================================================================
```

```
#include <pic.h>
#include "pic12f675.h"

// ======================================================================
// Configuration
// ======================================================================
__CONFIG(FOSC_INTRCIO & WDTE_OFF & PWRTE_ON & MCLRE_OFF & BOREN_ON & CP_OFF & CPD_OFF);
#define _XTAL_FREQ 4000000        // 4MHz

// ======================================================================
// [function] I2C I/F
// ======================================================================
#define SCL         GP0       // SCL=GP0
#define SDA         GP1       // SDA=GP1
#define SDATRSTRISIO1
// [I2C] Start condition
void i2c_start(void)
{
     SDA = 1;
     SCL = 1;
     __delay_us(5);
     SDA = 0;
     __delay_us(5);
     SCL = 0;
}
// [I2C] Stop condition
void i2c_stop(void)
{
     SDA = 0;
     SCL = 1;
     __delay_us(30);
     SDA = 1;
     __delay_us(10);
     SCL = 0;
}
// [I2C] Write data
void i2c_write(unsigned char data)
{
     unsigned char i;
     for(i=0; i<8; i++) {
          if(data&0x80) SDA=1;
          else SDA=0;
          data <<= 1;
          SCL = 1;
          __delay_us(2);
          SCL = 0;
          __delay_us(2);
     }
     SDA = 1;
     SDATRS = 1;
     __delay_us(2);
     SCL = 1;
     for(i=0; i<5 && SDA; i++)
          __delay_us(2);
     SCL = 0;
     SDATRS = 0;
}
// [I2C] Send data
void i2c_send(unsigned char madr, unsigned char padr, unsigned char data)
{
     i2c_start();
     i2c_write(madr);
     i2c_write(padr);
     i2c_write(data);
```

```
    i2c_stop();
}

// ====================================================================
// [function] Main
// ====================================================================
void main(void)
{
    unsigned short i=0;
    unsigned short adr=0xD2;

    // Initialize PIC
    CMCON    = 0b00000111;
    ANSEL    = 0b00000000;
    PEIE     = 0;
    TRISIO   = 0b00000000;
    GPIO     = 0b00000000;

    // Waiting for 500ms
    __delay_ms(500);

    // Change of the slave device address (0x10-0xEE -> adr)
    for(i=0x10; i<0xF0 ; i+=2)
        i2c_send(i, 0xF0, adr);

    // Initialize 40bar module
    i2c_send(adr, 0x02, 0x80);
    i2c_send(adr, 0x03, 15);
    i2c_send(adr, 0x04, 80);
    i2c_send(adr, 0x05, 40);
    i2c_send(adr, 0x06, 40);

    while(1){
        i2c_send(adr, 0x00, 40);
        __delay_ms(500);
    }
}
```

たとえば、上記プログラムでは新規アドレスとして 0xD2 **を指定しています。** (**変数名**:adr)
この値を、設定したいスレーブアドレスに書き換えてご利用下さい。

PIC12F675 **に上記プログラムを書き込み、当モジュールを繋いで通電すると、新しいスレーブアドレスが書き込まれ、一度再起動されます。 書き込みに成功すると、以降** 0.5 **秒毎に新しいアドレスに対して位置**=40 **を送信し続けるデモンストレーションを始めます。**(**自動減衰＋ピークバー付き**)

11. 点灯試験モード設定 (PointAddress = 0xF2)

当モジュールは、電源の接続時に全灯点灯試験を行います。 これは当モジュールのプロセッサに異常がないか、また断線等の不具合が起きていないかをチェックする為の動作ですが、この動作を OFF **にしたり、別の点灯方法を選択することができます。**

点灯試験モードの切り替えは、「10. カスタムスレーブデバイスアドレス設定」と同様の回路とプログラムで内部のフラッシュメモリを書き換える必要があります。前章のプログラムコードで簡単に説明すると、i2c_send(i, 0xF0, adr); の部分を i2c_send(i, 0xF2, 1);のように書き換えるだけです。

点灯試験モードは次のページの3種類があります。 (**出荷時は** 「1」 **が設定されています**)

設定値	動作	備考
0	点灯試験 OFF (※)	
1	アップダウン型点灯試験	※出荷時デフォルト
2	5段アップダウン型点灯試験	
3	単純な全灯 ON/OFF による点灯試験	

※設定値=0 (点灯試験 OFF) の場合、すぐに I2C の入力データの表示状態に入ります。
※点灯試験中も I2C の入力は受け付けています。 点灯試験中に送信したデータは点灯試験完了後に表示されます。

12. LED アレイユニット数変更指示 (PointAddress = 0xF4)

これまで説明した通り、当モジュールはプリント基板の切断加工により、20連化、または30連化に改造ができます。その際に、アドレス 0x02 の機能設定で「左右反転」を ON にすると、通常では位置 40 を基準点とて動作しますので、表示上の不都合が生じます。

そこで、この機能を使い、実装した LED アレイの数 (2～4個) を登録してあげる必要があります。
(デフォルトでは4個実装で「4」が登録されています)
登録方法は前章と同様、`i2c_send(i, 0xF0, adr);` の部分を `i2c_send(i, 0xF4, 2);`のように書き換えるだけです。

13. 機能設定例 (PointAddress = 0x02 - 0x06)

▼VU メータ風に利用したい場合は、次のように設定します。

0x02 = 0b10000000
0x03 = 10
0x04 = 120
0x05 = 40

自動減衰機能 ON。減衰速度は 2ms (200us×10)、ピーク保持時間は 24ms (200us×120)、ピークバー移動時間は 8ms (200us×40)。 あとは機能アドレス 0x00 に対して 0～40 の値を送信するだけで、ピークバー機能付きのバーグラフ表示になります。

▼通常のバーグラフとして利用したい場合は、次のように設定します。

0x02 = 0b00000000
0x03 = 15
0x04 = 0

自動減衰なし、アップダウンともに高速移動 3ms (200us×15)。0x03=0 でも構いませんが、アップダウンが激しい時に視覚的に追えなくなることがありますので、若干移動を入れると視認性が上がります。

14. 仕様

項目	内容	備考
電源電圧	DC5V±10%	安定電源必須（DC4.5V – 5.5V）
消費電流	1.8mA	無点灯時の消費電流
通信方式	I2C-BUS / 400kbps / 7bit	
表示方式	10 連 LED アレイ(※)	
点灯試験	起動時全灯点灯試験あり	OFF、パターン1・2・3の4種類から選択可能
寸法	105.0mm × 10.0mm	
重量	2g	モジュール基板のみの重量

※LEDアレイ、及びピンヘッダは付属しません。 別途ご用意下さい。

15. 免責事項

- 当モジュールは、専用に開発したチェックシステムにより、全てのモジュールに通電を行い、正常動作を確認した後で出荷していますが、不審な点が御座いましたらお問い合わせ下さい。
- 当モジュールは電子工作レベルの知識、更にマイコンプログラミングレベルの知識を持つエンジニアの方のご利用を想定しています。 技術的な質問にはお答え致しかねますので御了承下さい。
- 当モジュールを試用、使用または使用不能により生じた如何なる損害、損失についても、当方は一切の責任を負わず補償致しません。 自己責任でご利用下さい。
- ハンダ付けの工作ミス、適合外のLEDアレイを使用したことによる表示不良、実装方向のミス、端子の接続ミス、電源の逆接続等による破損、電源電圧の許容範囲を超える電圧を供給した場合の破損、物理的破損等、加工中、加工後に発生した不良は初期不良にはなりません。
- 本ドキュメントの無断使用、無断転載を禁じます。

16. 開発元・連絡先

KitaLab online store / 北ラボ
store@kitalab.com
http://store.kitalab.com/
第一版 2012/07/28