

第1章 電光掲示板の製作を通して理解する

LEDマトリクス・モジュールの制御と漢字表示のテクニック

◆開発環境
HEW3, FDT

山本 秀樹
Hideki Yamamoto

本章では、電光掲示板の製作を通して、LEDマトリクス・モジュールの制御と漢字表示の方法をマスターします。

製作した電光掲示板の構成を図1に示します。この電光掲示板は、表示する内容を文字列として保持し、またフォント・データも内蔵しています。文字列のそれぞれの文字に対応するフォント・パターンをスクロールしながら表示します。

使用するLEDマトリクス・モジュール(以下、LEDモジュール)は、16×32ドットのAD-501-B(入手先:秋月電子通商)です。これは、LEDドライバやシフトレジスタを内蔵しており、6本の信号線で16×32個のLEDマトリクス全体を制御できます。

付録マイコン基板 MB-H8 との接続方法

電光掲示板の回路図を図2に示します。4月号の付録マイコン基板 MB-H8 と LEDモジュールは、H8/3694FのI/Oポートと信号線で接続するだけです。

LEDモジュールの6本の信号線のうち、CLOCKはほかの信号と独立に制御するので、それを明確にする

ためにポート1に分けて接続しました。残りの5本の信号線のうち、今回の実装ではSTROBEは使用しないので、“L”に固定しています。それ以外の4本の信号線は、ポート5に接続しました。

LEDマトリクス・モジュールのしくみ

● 三つの16ビットLEDドライバで駆動される

図3に、LEDモジュールの内部回路を示します。三つの16ビットLEDドライバLC7932M(三洋電機)によりLEDモジュールを駆動しています。LC7932Mは、本来シフトレジスタの値を双方向にシフトできるのですが、AD-501-BではSINから入力しSOUTに出ていくように固定されています。

LC7932Mに値をセットするためには、16ビットのデータをCLOCKの立ち上がりでシリアルにSINから入力し、その値をラッチで保持する使い方になります。また、今回は使用していませんが、STROBEを“H”にすることにより、ラッチにセットされた値にかかわらず、強制的にLEDを消灯することもできます。

図4(p.114)に、LEDモジュールへの信号のタイ

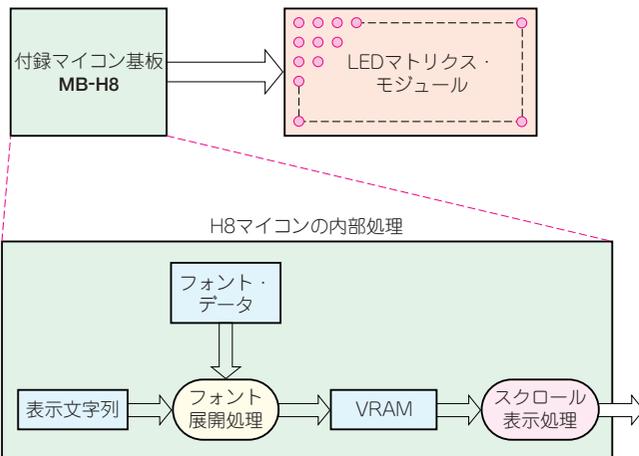


図1 製作した電光掲示板の構成

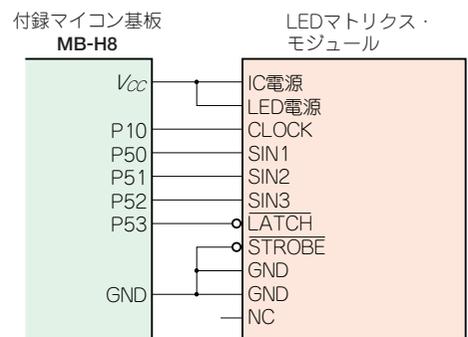


図2 電光掲示板の回路図

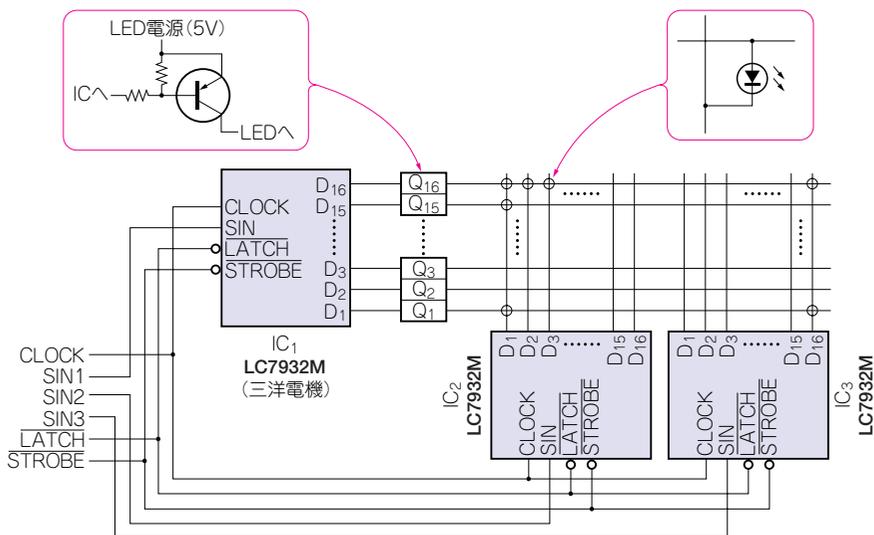


図3 LEDモジュールAD-501-Bの内部回路

ムチャート例と、その結果の表示を示します。この例では、SIN1のD15だけを“H”にしているのです。LEDモジュールの上から2行目が表示対象になります。SIN2はD16を“H”にしたので、LEDの左半面のうちD16が点灯します。同様に、SIN3はD2とD1を“H”にしたので、右半面のうちD2とD1が点灯します。

● ダイナミック表示

AD-501-Bのように、個々のLEDにドライバがあるのではなく、行と列のドライバから構成されている場合、任意のパターンを表示できるのは1行または1列単位です。

そのため、マトリクス全体に任意の文字や図形を表示するためには、1行または1列ずつの表示を高速に繰り返すダイナミック表示を行う必要があります。

AD-501-Bの場合は、ハードウェアの構成上、1行(横32個のLED)ずつ表示することが推奨されています。

図5(p.115)にダイナミック表示の例を示します。(a)のパターンを行単位にダイナミック表示する場合、ある瞬間を見ると(b)～(h)のような1行の表示になります。これを高速に繰り返すと、人間の目には(a)のように見えます。この説明にならえば、図4は時刻 $t+1$ の表示になります。

ソフトウェアの制作

● ソフトウェアの概要

製作した電光掲示板では、表示する文字列を文字コ

ードで保持します。また、LEDモジュールに表示するビット・パターンを一時的に保持するために、VRAMと名付けた配列を用意しています。

文字列を表示するときには、まず文字列内の表示する範囲について、それぞれの文字に対応するフォント・データをVRAM上に展開します。次に、VRAM上のデータをLEDモジュールに繰り返し出力し、ダイナミック表示を行います。

文字列が長すぎてLEDモジュールに表示しきれない場合、後述するスクロール表示を行います。スクロール表示を行うために、VRAMの大きさをLEDモジュールよりも大きくしています。各種変数の意味は稿末の図7にまとめています。

● フォント・データの形式

最初に、アルファベット、記号、半角カナといった1バイト文字からなる文字列の表示について解説します。フォント・データは配列ankFontに保持しています。それぞれの文字の大きさは 8×16 ドットであり、フォント・データは1次元のバイト配列です。

この配列内でフォントは、それに対応する文字の文字コード順に並んでおり、最初の文字コードのフォントはankFont[0]～ankFont[15]、2番目の文字コードのフォントはankFont[16]～ankFont[31]に格納されています。

文字コードから、それに対応するフォントのインデックスへ変換するために、FONT_ANK_INDEXマクロを用意しました。この変換処理はフォント・データに含まれる文字に依存するので、フォントごとに作成する必要があります。