プロローグ

液晶ドライバ内蔵 超低消費電力マイコン・モジュール誕生！

―― 手っ取り早く表示機能を組み込みたい人へ

本書には 72 ピクセル×32 ピクセルで表示が可能な液晶パネルと，液晶のコントローラを内蔵したセイコーエプソン製マイコン S1C17702 を搭載した基板が付属しています．この基板を題材に，本書では組み込み機器のプログラミングについて試しながら学びます．

● まずは動かしてみる

付属液晶マイコン基板には，出荷時にすでにプログラムが書き込まれています．第 1 章では動作確認を行うための方法について説明しています．本書を入手したなら，まず第 1 章を読んで基板を動かしてみましょう．

基板を動かすにあたり，はんだ付けや外付け電源などは不要です．USB ケーブルが 1 本あればまずは動作させることができます．

● 搭載マイコンの詳細は第 2 章，基板の仕様は第 3 章

そして第 2 章では，付属液晶マイコン基板で採用したセイコーエプソン製マイコン，S1C17702 について解説します．このマイコンは 16 ビット RISC CPU コアをもち，128 K バイトのフラッシュ ROM，

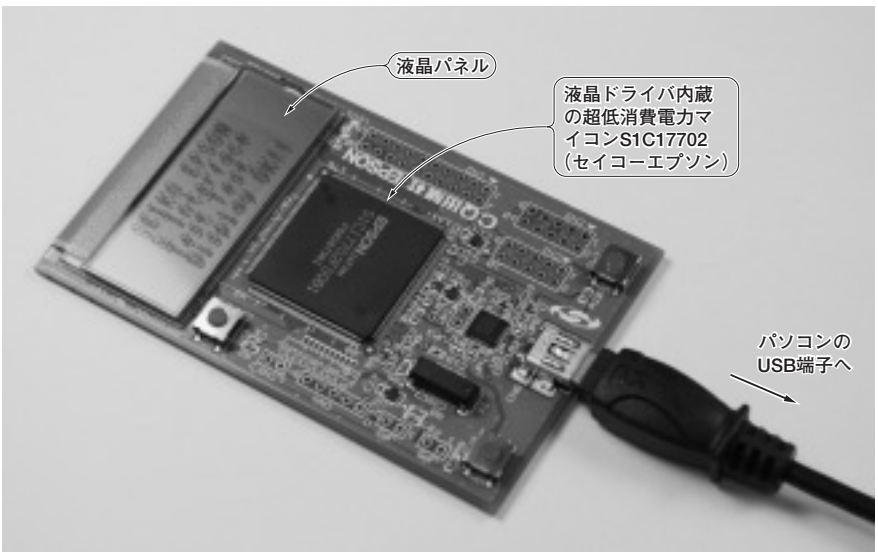


写真 1　開発中の様子

12 K バイトの RAM を内蔵したうえに液晶コントローラを内蔵しています．そのため，本基板ではマイコンに液晶表示装置を直結しています．

続く第 3 章では基板について解説をしています．もし，ハードウェアを増設したい場合にはこの章を参考にしてください．

● 開発環境のセットアップは第 4 章

第 4 章では付属液晶マイコン基板用のソフトウェアを開発するための開発環境をインストールします．開発環境は CQ 出版社 Interface の Web ページ，http://interface.cqpub.co.jp/および，そこから張られているリンクからダウンロードできます．これらダウンロードしたソフトウェアと，一般的な Windows パソコンがあればすぐにでも開発が行えます．

S1C17702 の開発環境は，統合開発環境として Eclipse を採用しているため，マウスを使って開発とデバッグが行えます．この Eclipse はマウスを使った GUI（Graphical User Interface）を提供するために使われており，C コンパイラとしては一般的な GCC（GNU Compiler Collection）を使用しています．また，デバッガには GDB（GNU Debugger）を使用しています．

GCC と GDB は今回採用した S1C17702 以外にも非常に多くの CPU に対応しており，地球上に存在するほぼすべて（？）の CPU 向けのプログラムを開発することができます．この付属液晶マイコン基板を使った開発で学んだ知識は，趣味に仕事に役立つことでしょう．

● 第 5 章～第 7 章でサンプル・プログラムを動作させる

第 5 章以降にはサンプル・プログラムを掲載しました．これらのサンプル・プログラムも Web ページからダウンロードし，コンパイルしてすぐに実行することができます．

第 5 章は液晶表示部のプログラミングです．この基板の特徴であり，世の中の電気製品に数多く搭載されている液晶パネルですが，それをどうやって操作し，任意の文字や絵を表示するのかを知っている人は少ないでしょう．そこでこの章のプログラムを動かしつつ，液晶表示部のプログラミングについて学んでいきます．うまく動作したら，文字や絵を少しずつ変えてみましょう．液晶表示部は，その効果が一目瞭然に現れるので，改造が成功したかどうかがすぐにわかります．

第 6 章は GPIO（General Purpose Input/Output）のプログラミングです．GPIO，すなわち汎用 I/O は CPU のピンであるディジタル入出力端子を使って外部回路を制御するものです．本基板では，出力が LED に，入力がスイッチにつながっています．第 6 章ではスイッチ入力を使って早押しカウンタを作成します．

第 7 章では UART（Universal Asynchronous Receiver Transmitter）を使います．俗に「シリアル」と呼ばれる一般的な通信方式です．UART を使えば，外部の機器をシリアルで制御できるほか，変数の値をシリアル経由で外部に表示させるようなデバッグ用途にも使えます．

● 第 8 章でゲームを作る

第 8 章は集大成としてブロック崩しを作ります．本基板には都合良く（？）プッシュ・スイッチが二つ付いています．このスイッチを使ってパドルを左右に動かし，ブロック崩しを動作させます．ここまでの知識があれば，ブロック崩しだけでなく，お好みのゲームを作れるようになるでしょう．

＊

液晶表示機能とマイコンを組み合わせた組み込み機器は非常にたくさんあります．つまり，それだけ需要があるということです．本書を通じて学んだことは，趣味に仕事に，即戦力として役に立つことでしょう．学ぶといっても「お勉強」ではなく，手を動かし，楽しみながら学んでいってください．

第1章 付属液晶マイコン基板を動かしてみよう！

―― USB ケーブルをつなげばすぐ動く！

本書に付属している液晶マイコン基板ははんだ付け不要，後付け部品なしで動作させることができます．まずは，動作確認をしてみましょう．また，コネクタなどをはんだ付けするとデバッグが容易になるなどの利点があります．

1-1 付属の液晶マイコン基板の仕様

本書にはセイコーエプソン社の S1C17702 マイコンを使用した基板が付属しています．

このマイコンは C17 というセイコーエプソン社のオリジナル CPU コアを使用しています．C17 マイコンは，ハーバード・アーキテクチャを採用した 16 ビット RISC CPU で，プロセッサの使用するフットプリントが少ないという特徴をもっています．

付属液晶マイコン基板のパッケージが大きいのは，おもにドット・マトリクスを駆動するためのコントロール信号が直接出力されているためです．104 本もの信号線が液晶パネルに直結されているのです．

付属液晶マイコン基板に使用されている S1C17702 マイコンは 176 ピンのパッケージなので，LCD コントローラ信号を除くと 72 ピン，物理的に大きなパッケージを採用しているためか，GND や電源に多めのピンを割り当ててある（電源，N/C ピンだけで 35）ので，それらを最小限の各 1 ピンずつとすれば，39 ピンとなります．

I/O ピン数 39 ピン程度の 1 チッププロセッサ LSI はほかに，PIC や AVR，8051 などがありますが C17 コアは内部の回路規模を 8051 よりはるかに小さいという特徴があります．

付属液晶マイコン基板では QFP パッケージの LQFP21-176pin 版が使用されていますが，BGA バージョンの FBGA8H-181 というものもあります．

さて，細かな設定や理屈はさておき，早速，付属液晶マイコン基板が正しく動作するものであることを確認しましょう．

1-2 動作確認

付属の液晶マイコン基板は「いっさいの後付け部品なしに動作可能」なように設計されています．電源は USB - シリアル変換チップである CP2102 を通じて供給されるので，パソコンのほかには外部電源などは不要です．パソコン（パソコンでなくともバス給電機能をもつ USB ポートなら実は何でもよい）と基板とを接続する，USB の A-miniB ケーブル（**写真 1-1**）があれば十分です．USB の miniB ケーブルは比較的多くの USB 機器で使用されているので，慌てて買いに行く前に身の回りを探してみることをおすすめします．数年前に購入した USB 外付けの DVD ドライブのケーブルなどが使えたりするものです．もし，

写真 1-1　パソコンと付属基板を接続する USB A-miniB ケーブル

ケーブルの購入に秋葉原などに出かける場合には，後述する「1-4 より楽しく遊ぶために」で挙げている部品も一緒に買ってくることをおすすめします．

1-3　デモ・プログラムを使って動作を確かめる

付属基板にはデバッグ支援と動作検証を目的に，GDB のバックエンドと，簡単なデモ・プログラムがあらかじめ書き込まれています．後付部品を取り付ける前にデモ・プログラムを使って付属液晶マイコン基板の動作確認をしておきましょう．

● USB ブリッジ IC CP2102 ドライバをインストールする

付属基板をパソコンに接続する場合，パソコン側に CP2102 のデバイス・ドライバをインストールしておきます．このテストではパソコンの USB ポートを単なる＋5 V 電源として利用するのでドライバのインストールは必須ではありませんが，あとあと必要になるのでここでインストールをしておきましょう．

デバイス・ドライバは CP2102 の製造元である Silicon Laboratories 社の Web ページからダウンロードできます(**図 1-1**)．「CP2102 download」などで検索するとトップに表示されるでしょう(日本語だけでなく，全言語から検索すること)．

● デモ・プログラムの起動

付属基板とパソコンを USB ケーブルで接続します(**写真 1-2**)．しかしそれだけでは何も起こりませんが，故障ではありません．電源が接続されただけの本機は，デバッガとの接続を待っている状態にあります．

この状態で，スイッチ SW_1(リセット)と，SW_2 を同時に押し，SW_1 を離すとデモ・プログラムの実行が始まります(**写真 1-3**)．独自のアプリケーション開発中の，デバッガを経由しないアプリケーションの起動方法も同様です．

あるいは，ジャンパ JP_2 をショート状態 (PC などについているジャンパ・ブロックでショートします)．で電源を投入するとやはり，デバッガではなくアプリケーションが立ち上がるようになっています．

デモ・プログラムが立ち上がると，本機の LCD には**写真 1-4** のような画面が表示されます．画面の表示に従って SW_2 と SW_3 を押すとメッセージが切り替わるので，SW_2 および SW_3 の動作を確認することができます(**写真 1-5**)．

なお，次章以降に示す手順に従って自分の作ったプログラムを本機にロードすると，同じ手順でユーザ・プログラムが走り出すので注意してください．特に，本章を読み飛ばしたあと，うまく動かないからという理由で本章に戻ってきた人は要注意です．

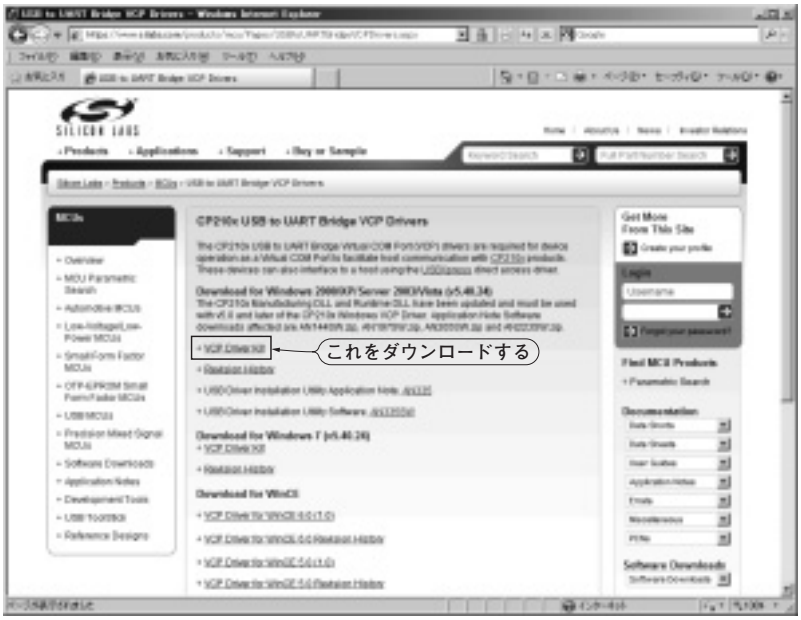


図 1-1　付属基板に搭載された USB ブリッジ IC CP202 のメーカ Silicon Laboratories 社の Web ページからデバイス・ドライバをダウンロードする

https://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx

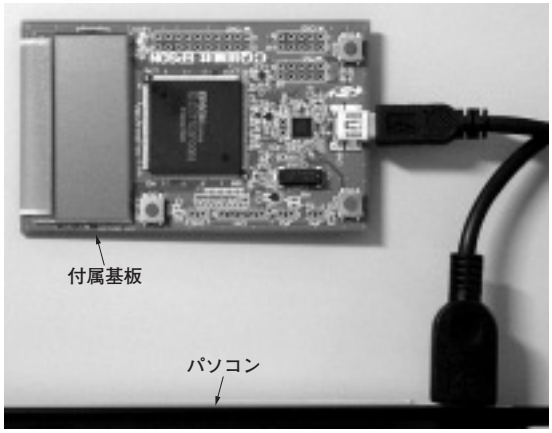


写真 1-2　付属基板とパソコンを接続する

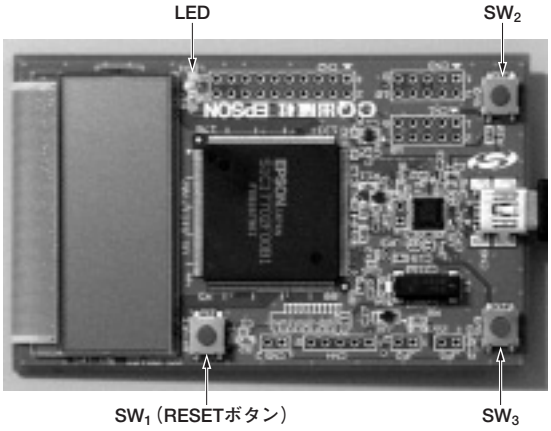


写真 1-3　SW_1 と SW_2 を押し，SW_1 を離す

写真 1-4　デモ・プログラムが立ち上がった画面

写真 1-5　SW_3 を押すように指示される

1-4　付属基板のカスタマイズ

S1C17702 プロセッサは，C17 というセイコーエプソン社のオリジナル CPU をコアにしています．このコアは 8051 よりも小さいので，余ったスペースにというわけではないのでしょうが，さまざまな周辺回路が入っています．標準状態でもいろいろなことができますが，多少の外付け部品を付けるとさらに多くのアプリケーションを作ることができます．

● 入出力の取り出し用コネクタなどを取り付ける

まずは，2 列のストレートピン・ヘッダを 5 本ずつ切り取り，CN_1 と CN_3 にはんだ付けしておきましょう．また，残った部分で 2×1 ピンのコネクタを作り，JP_1，JP_2，JP_5 にそれぞれ取り付けます(**写真 1-6**)．

内蔵の簡易デバッガではなく，純正のデバッガを使用する場合にはシングルラインのピンを CN_4 に取り付けます．筆者は手持ち部品の関係で丸ピンの IC ソケットに使用するピンを使用しましたが，普通の角ピンのヘッダのほうがよいようです．

CN_2 には，C17 コアの最初のブート・ブロックを保持しているフラッシュ ROM の内容を壊してしまったときのために，メスのピン・ソケットを裏側に取り付けます(**写真 1-7**)．CN_2 に出ている信号線も多数あるので，それらの I/O を使用するときは親基板のような形で使用するのがリーズナブルだと思います．

● 超低消費電力動作を可能にする方法

本機は基本状態で 8 MHz の OSC3 クロックと 2.5 MHz の内部オシレータによるクロックを CPU および周辺ロジックで使用可能ですが，X_2 と書かれた未実装パターンに 32.768 kHz のクリスタルと，その隣

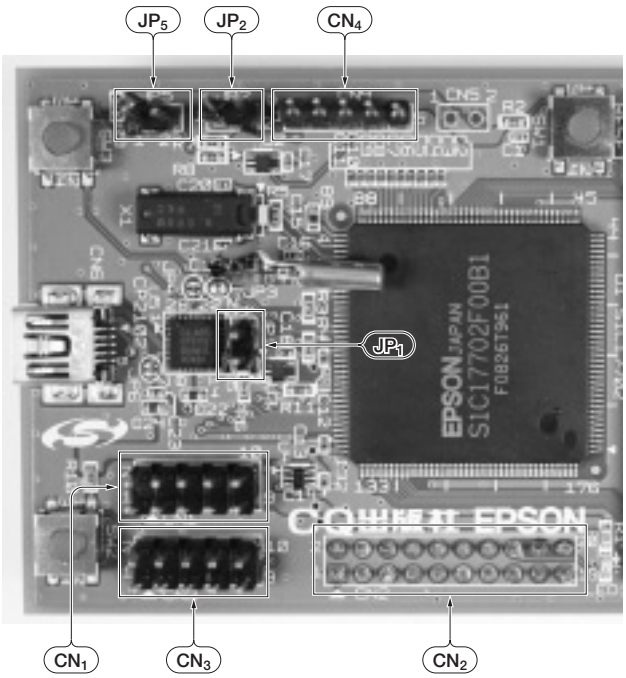


写真 1-6　コネクタとピン・ヘッダを取り付ける

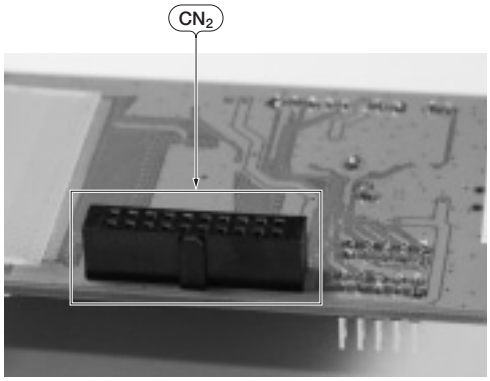


写真 1-7　CN_2 にはメスのピン・ソケットを付ける
(基板裏面)

第4章 搭載マイコンの仕様とアーキテクチャ

―― 72 ピクセル×32 ピクセル液晶を直接駆動！ 動作時 9μA の超低消費電力！

本章では，付属基板に搭載されている液晶ドライバ内蔵の 16 ビット・マイコン（S1C17702，セイコーエプソン）マイコンのもつ機能や性能を説明します．S1C17 という名称の CPU コア以外に，各種の機能回路を内蔵しています．

4-1 搭載マイコンに内蔵された CPU と周辺回路

■ スペック

付属基板に搭載されているマイコン（S1C17702）は，16 ビット RISC CPU 以外に次のような機能回路を内蔵しています．

- 128 K バイトのフラッシュ ROM
- 12 K バイトの RAM
- シリアル・インターフェース（IrDA1.0 対応の UART，SPI，I^2C）
- 8 ビット・タイマ，16 ビット・タイマ，PWM & キャプチャ・タイマ，計時タイマ，ストップウォッチ・タイマ，ウォッチドッグ・タイマ
- 28 本の汎用入出力ポート
- 最大 72 セグメント×32 コモン（72 ピクセル×32 ピクセル）の液晶ドライバ
- 電源電圧昇圧回路
- 電源検出回路（SVD：Supply Voltage Detector）
- 32 kHz と最大 8.2 MHz の発振回路
- 1.8 V 出力の定電圧回路

CPU は，RISC 処理によって 1 命令を 1 クロックで実行し，供給する電源電圧が 1.8 V のときでも，8.2 MHz で動作させることができます．コプロセッサを内蔵しており，乗算と積和演算機能をもっています．

図 4-1，**図 4-2**，**表 4-1**，**表 4-2** に内部ブロック図，仕様，端子説明を示します．

■ メモリ空間

図 4-3 に搭載マイコンに内蔵されているメモリ・マップを示します．

● フラッシュ・メモリ

0x8000 ～ 0x27fff 番地の 128 K バイトの領域は，プログラムやデータを書き込むフラッシュ・メモリです．0x8000 番地は，ベクタ・テーブル・ベース・アドレスとして定義されているため，この領域の先頭にベクタ・テーブルをおく必要があります．

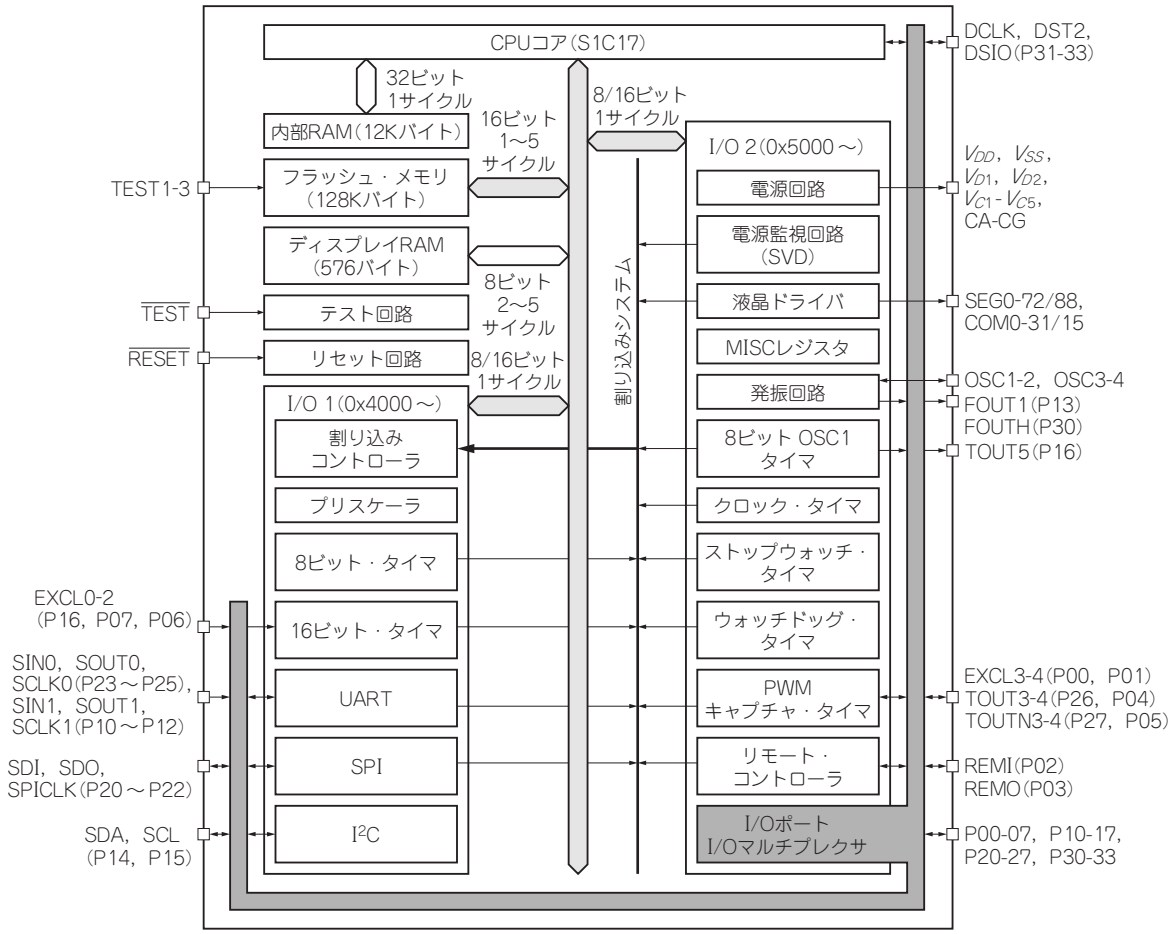


図 4-1　搭載マイコン (S1C17702) のブロック図

● RAM

0x0 ～ 0x2fff 番地の 12 K バイトの領域は RAM です．変数を格納したり命令コードをコピーすれば，高速に処理することができます．

● 表示 RAM

0x80000 ～ 0x8055f 番地の 576 バイトの領域には，内蔵液晶ドライバ用の表示 RAM が 8 ビット・デバイスとして割り付けられています．表示に使用しない領域は，汎用 RAM として使えます．

● 内部周辺回路エリア

0x4000 番地から始まる 1 K バイトと 0x5000 番地から始まる 4 K バイトには，I/O と制御レジスタが割り付けられています．

■ CPU 周辺の機能回路 (ペリフェラル)

● 入出力ポート

信号を入出力するための端子です．全部で 28 本あり，ソフトウェアによって入出力方向を切り換えられます．一部を除いて，内部周辺モジュールの入出力端子を兼ねていますが，周辺モジュールに使用しない端子は，汎用の入出力ポートとして使えます．P0 ポートと P1 ポートは入力割り込みを生成できます．

第5章 LCDのコントロール・プログラミング

―― 液晶画面に自由に絵や文字を描こう！

本付属基板の最大の特徴は液晶画面です．ここでは，液晶画面に絵や文字を描画する方法について解説します．サンプル・プログラムで動作確認をするだけでなく，すでにある描画ルーチンをそのまま自分のプログラムに組み込むことも可能です．

5-1　S1C17702に内蔵されたLCDコントローラについて

本書の付属基板に採用しているS1C17702プロセッサには，LCDコントローラが搭載されています．付属基板には横72ピクセル，縦32ピクセルのLCDが使用されていますが，LCDコントローラには横87ピクセル，縦16ピクセルのLCDを接続することもできます．この場合には，フレーム・バッファRAMを2ページぶん保持できるので，次に表示する内容をあらかじめ次のフレーム・バッファに用意しておき，描画終了のタイミングで表示するアドレスを切り替えることにより，描画のちらつきを防止するというテクニックも使えます．

プロセッサとLCDは，コモン(以下，COM)信号と，セグメント(以下，SEG)信号とがマトリクスを構成するように接続されています(**図5-1**)．そのため，72×32ピクセルの表示を行うためには，104本の信号線を接続する必要があります．

表5-1に示すように，VGAの信号線11本(ピン数は15本だがN/C 3本を除き，2本あるGNDを1本にまとめた場合)に比べると圧倒的に多い配線です．これは，VGAがもともとはテレビなどに使われていたラスタ・スキャン型のブラウン管に信号を送るために設計されたためです．VGA信号とモノクロ・テレビの信号を**図5-2**に示します．周波数の違いさえ気にしなければ，VGAからモノクロNTSCフォーマットの信号を作る回路はきわめて簡単に作ることができます．

LCDはブラウン管に比べると，「軽い」，「薄い」，「消費電力が小さい」などいろいろな利点がある反面，「反応速度が遅い」という欠点を今なおもっています．これは，ブラウン管が電子ビームの軌道を電磁石で曲げ，ラスタ状にスキャンする間に電子ビームの強さを変えるという原理を活用しているのに対し，LCDは「液晶」という液状の結晶に電圧をかけたりかけなかったりして，結晶の方向を制御して表示を行うという，見るからに遅い原理を活用しているためです．

S1C17702に搭載されたLCDコントローラは，**図5-3**のような信号を生成します．VGA信号にせよ，LCD信号にせよ，CPUが独自に作り出すのは「非常に面倒な信号」であることがわかると思います．

と，比較的長い前振りを書いたのは，LCDコントローラの初期化が比較的面倒な作業だからです．と言っても，一度書いてしまえば，あとはそのまま使い回せばよいので，読者はサンプル・プログラムを「コピペ」すればよいと思います．ただし，動作を少し変えたいとか，コードを100％理解しないと使いたくないという人は，以降の説明をお読みください．

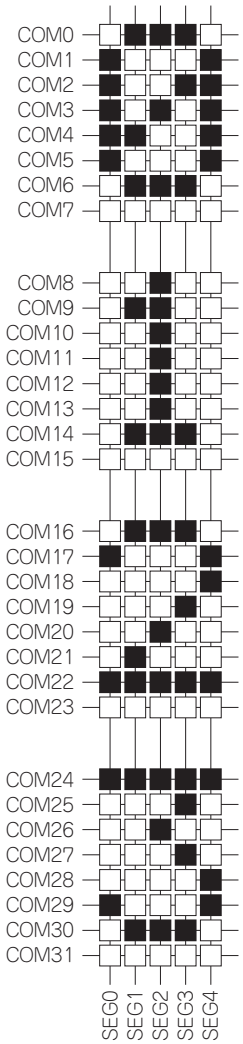
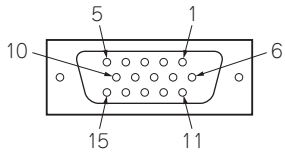


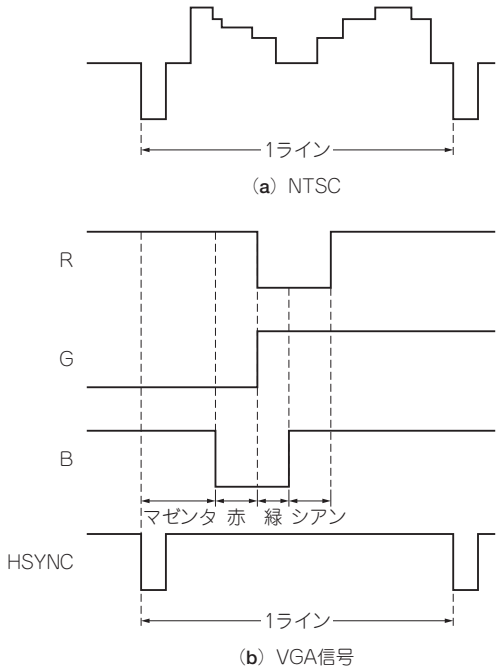
図 5-1　プロセッサと LCD との接続

表 5-1　VGA の信号線とピン配置

ピン番号	名　称
1	RED(赤)
2	GREEN(緑)
3	BLUE(青)
4	N/C
5	GND
6	RED_RTN
7	GREEN_RTN
8	BLUE_RTN
9	N/C
10	GND
11	N/C
12	SDA
13	HSYNC or CSYNC
14	VSYNC
15	SDL



VGA コネクタ



(**a**) NTSC

(**b**) VGA信号

図 5-2　VGA 信号とモノクロ・テレビの信号

「コピペするからいい」という人は，5-2 節を読み飛ばしてもかまいません．

5-2　S1C17702 用 LCD ドライバ制御プログラムの設計

まずは，プロセッサのマニュアルと回路図をにらめっこしてみましょう．人の作ったものには必ず間違いがある可能性を考慮すると，実際にデバッガを接続し，レジスタ値を直接操作するという実験を行ってみるのもよいかもしれません．

ここでは，「本来の設計方法」というより，筆者が実際にたどった実装方法を順に説明しようと思います．まず，スタートポイントは，付属基板に USB ケーブルでパソコンと接続し，Insight (gdb) デバッガで，

第8章 ブロック崩しゲームを作ってみよう

―― 小型端末といえば，まずゲーム！

本章では，付属基板のLCD，I/Oポート，タイマ割り込みの各機能を使って簡単なゲームを作ってみます．ただし，付属基板にはボタン二つとLEDしか付いていないので，I/Oポートといっても別の部品を付けないかぎり，市販のゲーム機のように上下左右の操作やアナログ・ジョイスティックを使った操作，サウンドなどの機能はありません．特に，入力がボタン二つに限られているため，2ボタンで操作ができるゲームに限られてしまいます．

最近は，携帯電話などで遊べるワンボタン・ゲームが花盛りですが，さすがにモノクロで72ピクセル×32ピクセルのLCDでは，できるゲームも限られてきます．ここでは，左右のボタンだけで操作できるブロック崩しを作ってみましょう．

8-1　ブロック崩しプログラムの構想

最初に，プログラムをどういうふうに作るかを考えます．現在のGUI環境に慣れた人なら，イベント・ドリブンのプログラムをすぐに思いつくかもしれません．たとえば，キーボード入力によるイベントでパドルを左右に動かし，タイマ・イベントでボールを移動させるという方法です．

しかし，本書の付属基板にはこのような高度なOSは組み込まれていませんし，OSと呼べるような機能は，フラッシュROMやRAMの容量を考えても現実的ではありません．すなわち，勝手にイベントを

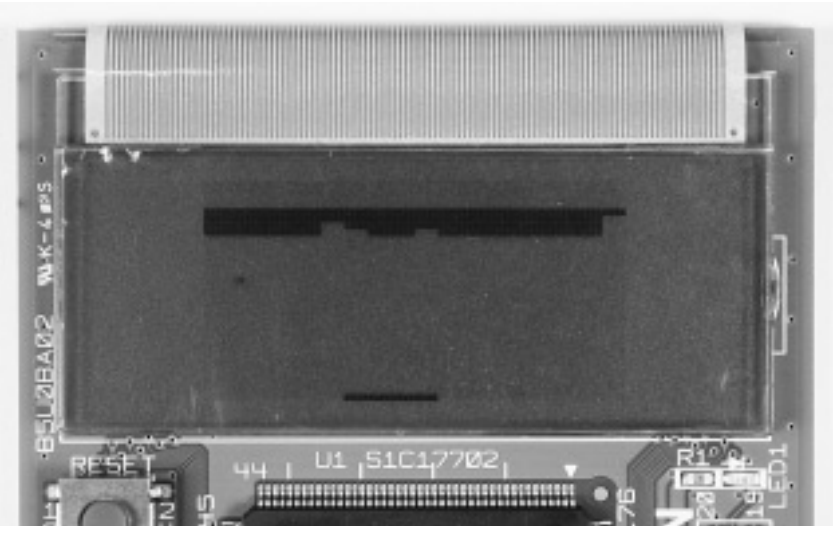


写真8-1　ブロック崩しゲームを作ってみた

流してくれるような仕組みは用意されていませんし，マルチタスクを実現するような機構も組み込まれていません．

とはいえ，第 4 章で紹介したように，S1C17702 プロセッサでは数多くのタイマが使用できます．また，第 5 章で紹介したように，LCD の割り込みも使用することができます．そこで，本章のプログラムはこれらの定期的な割り込みを使用した，疑似マルチタスクで実装してみたいと思います．

写真 8-1 が完成したブロック崩しです．

8-2　ブロック崩しプログラムの構造

本章で作成するプログラムは，大きく分けて四つの部分から成り立っています．

(**1**) 初期化部分
(**2**) ボールの移動を行う部分
(**3**) パドルの移動(ボタンによる操作)を行う部分
(**4**) 画面の描画を行う部分

初期化部分を除いて，それぞれの部分を独立したタイマ割り込みで駆動することにより，速度を独立にコントロールすることができます．つまり，ある機能の動作を速くしたければ，そのタイマの周期を速くすればよいわけです．そうすることによって，1 回の処理で変化させる量を固定でき，処理の簡略化が図れます．

それぞれの処理を擬似コードで書くと，(**1**)の初期化部分は**リスト 8-1**，(**2**)のボールの移動を行う部分は**リスト 8-2**，(**3**)のパドルの移動(ボタンによる操作)を行う部分は**リスト 8-3**，(**4**)の画面の描画を行う部分は**リスト 8-4** になります．

リスト 8-1　初期化

```
main()
{
    基本的な初期化
    ゲームデータの初期化
    各割り込み処理の登録
    タイマの起動
    無限ループ(ここでは何もしない)
}
```

リスト 8-2　ボールの移動

```
ball_process()
{
    ボールの位置を場合
    ブロックに衝突した場合{
        ブロックを消去
        ブロックがなくなったら{
            勝ち
        }
        ブロックによる反射
    }
    壁に衝突した場合{
        壁で反射
    }
    画面下側に出た時{
        パドルに当たったら{
            パドルでの反射
        }当たらなかったら{
            ボールロスト
            ボールがなくなったら{
                GAME OVER
            }残っていたら{
                次のボールで継続
            }
        }
    }
}
```

リスト 8-3　パドルの移動

```
button_process(){
    右ボタンを押されていたら{
        パドルを右に移動
    }
    左ボタンを押されていたら{
        パドルを左に移動
    }
}
```

リスト 8-4　画面の描画

```
dispay(){
    ブロックを描画
    パドルを描画
    前のボールを消去
    ボールを描画
}
```