第3章 付属マイコン・メーカ純正の無償開発環境で実行プログラムを作る

―― HEW をインストールして LED を点滅させる

本章では，H8SX/1655 マイコン・メーカ ルネサス テクノロジ製の統合開発環境 HEW（High-performance Embedded Workshop）を使ってプログラムを作り，実行形式のファイルを生成（ビルド）します．

開発環境 HEW を使う前の確認ごと

● 無償評価版 HEW の扱いについて

（1）問い合わせやサポートはお受けできません

本誌で使用する，無償の評価版ソフトウェア HEW については，ルネサス テクノロジからのサポートはいっさい受けることができません．

（2）ビルド後 60 日でオブジェクト・コード・サイズが 64 K バイト以下になります

無償評価版 HEW に組み込まれている H8SX 用コンパイラは，最初のビルドを始めてから 60 日間までは有償製品版がもつ機能をすべて使えますが，60 日以降は，プログラムをビルドしてできるオブジェクト・コードのサイズが 64 K バイト以下に制限されます．

（3）HEW 単独では，ターゲット基板のデバッグはできず，有償のエミュレータ E10A-USB が必要です

本誌では，次章以降で，使用期限のないオープン・ソースの無償ツールを使って統合開発環境と強力なデバッガ機能を提供します．必ずしも，本章で解説する環境を構築する必要はありませんが，知識としては重要ですので，本章の内容を実際に確かめてみることを推奨します．

E10A-USB によるデバッグの方法は，付属 CD-ROM に収録された E10A-USB の使用方法.pdf を参照してください．

システム機器を開発するため，使用するマイコンの開発環境を取り揃えようと思ったとき，メーカ純正品とサード・パーティ品，および GNU GCC に代表されるオープン・ソース品のいずれかから選択することになります．

このうち，メーカ純正品は，CPU やデバイスの特徴を最もよく理解して設計されており，とくにオープン・ソースの GNU GCC に比べると，生成するコード・サイズが小さく，かつ実行速度も速いことが多いです．また多くの顧客が採用している実績から，生成コードの信頼性も高いものになっています．このため，実際の機器を開発する人は，メーカ純正品を選択する場合が多いです．

統合開発環境を使った開発の流れ

● ファイルやソフトウェアがメイン・ウィンドウ上で一貫して処理できるので便利

統合開発環境とは，プログラム・ソース・コードの編集から，実際のビルド（コンパイルやリンク）およびデバッグまでを一つのメイン・ウィンドウ上で一貫して処理できるプログラム開発ツールのことです．

統合開発環境の上で一つのプログラムを開発する際，プロジェクトまたはプロジェクト・ワークスペースという概念があります．この中には，そのプログラムのビルドに使うソース・コード類の格納場所，コンパイル時のオプション指定の方法，リンク時のオプション指定の方法，ライブラリの格納場所，最終的な

第4章 無償ツールでプログラムのビルド環境を作る

―― 統合開発環境 Eclipse をベースにして
使い勝手のよいツールを整備

本章では，H8SX/1655 のプログラム開発用にオープン・ソースの無償ツールによる統合開発環境を構築して，プログラムの作成から，ビルド，および実際の動作確認までを説明します．実際のプログラムのしくみや，ソース・レベル・デバッグの方法については次章以降で解説します．

■ プログラムやデータ類を事前に準備する

本誌に付属する CD-ROM 内のフォルダ「CQ」をコピーして，C ドライブ以下に，「C:¥CQ¥H8SX_1655¥...」という階層になるように置いてください．本誌では，この場所にプログラムやデータ類が置かれていることを前提として説明します．付属 CD-ROM に収録された開発環境のプロジェクトの設定は上記のデータ位置を前提にしたものになっているので，本誌の内容をトレースする場合は，上記位置にデータを置いてください．

オープン・ソースの無償ツールのメリット

● 技術情報がインターネット上にたくさん存在

マイコンのプログラム開発に使われるオープン・ソースの無償ツールとして代表的なものに，GNU GCC（GNU Compiler Collection）があります．

こうしたツールは無償で使えるというメリットに加えて，技術情報がインターネット上にたくさん存在しているというメリットもあります．ある問題に遭遇したとき，そのエラー・メッセージなどでインターネット検索すると，同じトラブルを経験した人の解決策を見つけることができることが多いのです．

本章以降は GNU GCC をベースに解説します．

● H8 以外の CPU のプログラム開発にも使える

それでも，手軽に強力な機能をすぐに使えるメリットは大きく，本誌のようにマイコンを学習していくためのツールとしては最適なものです．

また，このツールは，H8 系以外の CPU 向けにも多く開発されているので，1 回学習してしまえば，ほかの CPU のプログラム開発にもそのまま知識を応用できるので，習得して損をすることはありません．

● 信頼性は保証されていない

その一方で，メーカ純正品に比べると，生成するコードのサイズが大きく，また実行速度も遅い傾向があります．

さらに生成オブジェクトの信頼性は保証されていないので，高信頼性を要求される量産機器に採用するには注意が必要です．

主な三つのソフトウェア

本誌で構築する統合開発環境の全体構成を**図 1** に示します．また，各ツール間の連携方法を**図 2** に示します．各ツールはすべて無償で入手できます．

この統合開発環境によって，MB とパソコンとの間を USB ケーブル 1 本で接続するだけで，プログラムの開発・ビルドから，ダウンロード，ソース・レベル・デバッグまで行うことができます．

● 統合開発環境 Eclipse ：エディタやコンパイラ，デバッガを連携してくれる

ソース・コード・エディタやコンパイラを含む各ツールを有機的にまとめるソフトウェアを統合開発環境（IDE ： Integrated Development Environment）といいます．統合開発環境としては Eclipse を使用します．Eclipse はもともと Java の開発環境として IBM が開発しました．Eclipse 自身も Java のランタイム環境上で動作します．Eclipse は拡張性が高く，Java 以外に C/C++ などの開発環境としても広く活用されて

第7章 USB経由でパソコンと通信する

—— コンソール通信とバイナリ・データ転送を実現

本章では，付属基板（MB）とパソコンとの間で，USB経由で通信したりバイナリ・データを転送したりします．

● 付属基板で利用するUSB通信の方法

本誌で使用するUSB通信の方法を**表1**にまとめました．

H8SX/1655内蔵のUSBファンクション・モジュールは，USB 2.0フル・スピード（12Mbps）に対応しています．USB転送のクラスとしては仮想COMポートを使います．パソコン側のドライバは，第6章でGDBを立ち上げたときにインストール済みです．

本誌ではUSB通信そのものの低レベルなプロトコルの詳細説明は省きます．必要があれば参考文献やUSB関連のそのほか文献を参照してください．本誌では，USBをその詳細を知らなくても使えるソフトウェア環境を提供します．

Windows標準のソフトウェアで通信する

本誌で提供している各プログラムのプロジェクト内では，すでにprintf()やscanf()をサポートしているので，簡単にパソコン上のターミナル・ソフトウェアとコンソール通信が可能です．

● src/config.hを修正，ビルドして書き込む

各プロジェクト内のsrc/config.hをUSBで通信する場合は**リスト1**（**a**）のように，RS-232-C（SCI4）経由で通信する場合は**リスト1**（**b**）のように修正してビルドします．

RS-232-C方式の場合に，H8SX/1655内のSCIモジュールとしてはSCI4を使用していますが，これは内蔵フラッシュ・メモリのブート書き込みにも使用できるチャネルでもあるからです．

実際にパソコン上のターミナルと付属基板（MB）の間を**図1**のようにUSB経由で通信してみましょう．

表1　実験で使用するUSB通信モード

項　目	内　容	備　考
USB規格	USB 2.0 フル・スピード（12Mbps）	—
クラス	仮想COMポート	—
パソコン側ドライバ	C:¥CQ¥H8SX_1655¥Renesas¥USB_Driver	第6章（GDB）でインストール済

リスト1　コンソール通信をするための準備
src/config.hを書き換えてビルドする

```
//#define CONFIG_GDB_STUB_USB_ON
```
（コメント化する）

（**a**）USBで通信する場合

```
#define CONFIG_GDB_STUB_USB_ON
```
（アクティブにする）

（**b**）RS-232-C（SCI4）で通信する場合

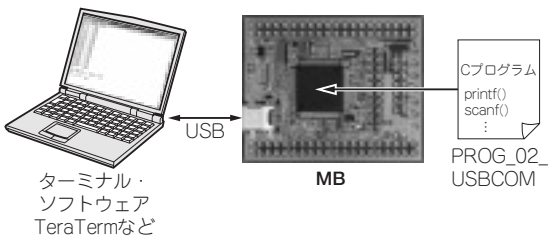


図1　ターミナル・ソフトウェアによるテキスト通信の実験
付属基板にプログラムPROG_02_USBCOMを書き込み，パソコン上のターミナル・ソフトウェアと通信する

第8章 学習用の拡張基板を作る

―― ディスプレイへの文字表示から SD カード活用まで

本章では，付属基板（MB）を搭載できる学習用の拡張基板（SB）を作ります．ソフトウェアで制御する方法は第 10 章以降で解説します．

付属基板（MB）だけでもマイコンの学習は十分にできますが，腕に自信のある方は拡張基板を自作してオリジナルのシステムを開発に挑戦してください．

付属基板（MB）と接続して機能を拡張できるる 2 種類の基板を設計しました．

（1）MB を搭載してすぐに H8SX/1655 の使い方の学習を始められる拡張基板 **SB**（System Board）

キャラクタ LCD モジュール，アナログ入出力，時計機能，SD カード・ソケットなどを搭載しています．

（2）タッチ・パネル入力付きグラフィック LCD を取り付けて，MB を搭載することで表示制御が可能になる拡張基板 **TB**（TFT LCD Panel Board）

タッチ・パネル付きの TFT カラー LCD パネルを搭載することができ，付属基板（MB）と組み合わせることで，グラフィック・タイプの入力表示システムを開発できます．詳細は第 9 章を参照してください．

これら 2 種類の拡張基板が，自作の参考になるでしょう．時間に余裕のない方は，SB と TB（第 9 章参照）の完成品（マルツパーツ館扱い）を購入してください．

搭載されている機能

拡張基板 SB は，キャラクタ LCD モジュール，アナログ入出力，時計機能，SD カード・ソケット，RS-232-C インターフェースなどを搭載する機能拡張ボードです．

写真 1 に SB の外観を，**表 1** に仕様を，**図 1** に全体構成を，**図 2**（p.94 ～ 95）に回路図を示します．SB の完成状態の外観は，後出の**写真 3** を見てください．

● 16 文字× 2 行の文字表示ディスプレイ

SB 上には，SC1602（Sunlike Display Tech 社，以下 SUNLIKE）互換のキャラクタ LCD モジュール（16 文字× 2 行）を搭載できます．

図 3 に接続方式を示します．キャラクタ LCD モジュールは，H8SX/1655 の外部バス空間（エリア 2）に接続します．LCD モジュールのデータ幅は，初期化時に 4 ビットまたは 8 ビットから選択できますが，ここでは 8 ビットに設定して使います．

キャラクタ LCD モジュールは 5V 電源で動作するので，LCD モジュールからマイコンに向かう信号（リード・データ）のレベルを，5V から 3.3V にシフトする必要があります．そこで，データ・バスに TTL バッファ（U_3，SN74LVCZ245APW，テキサス・インスツルメンツ）を挿入します．

LCD モジュールのアクセス・タイミングは往年の 6800 系バスでありE信号が必要です．そこで，LCD モジュールに対する H8SX/1655 のアクセス方式を，バイト制御 SRAM インターフェースに設定し，チップ・セレクト $\overline{CS2}$ とロー・バイト・ストローブ $\overline{LLB}$ の入力負論理 AND（U_2，SN74LVC1G02DBVR，NOR ゲート，テキサス・インスツルメンツ）でE信号を生成するようにしました．

バイト制御 SRAM インターフェースに設定するため，エリア 2 は 16 ビット空間にします．その下位 8 ビット側に LCD モジュールを接続しています．エリア 2 はビッグ・エンディアン設定にするので，ソフトウェアが LCD モジュールをアクセスするときは，常にその空間の奇数側をバイト・サイズでアクセスします．

LCD モジュール内には，インストラクション・レジスタ（IR）とデータ・レジスタ（DR）の二つのレジスタ

第14章 アナログ信号のA-D変換とD-A変換の実験

―― H8SX/1655の内蔵A-DとD-Aを動かしてみる

本章では，付属基板(MB)に搭載されたH8SX/1655マイコンに内蔵された10ビットA-D変換器と10ビットD-A変換器を動かしてみます．H8SX/1655マイコンの使い方の学習を始められる拡張基板SB(System Board，第8章参照)を使ってアナログ信号を入出力します．

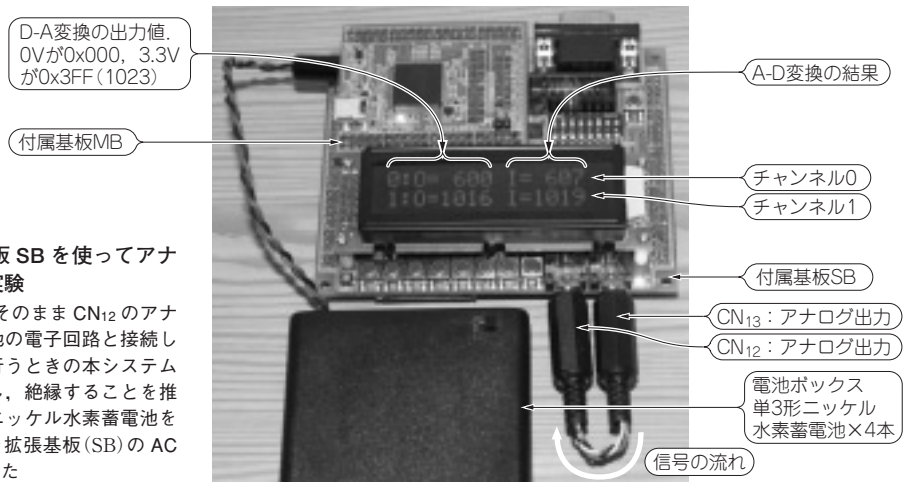


写真1
第8章で設計した拡張基板SBを使ってアナログ入出力プログラムを実験
CN_{13}からのアナログ出力をそのままCN_{12}のアナログ入力に接続している．他の電子回路と接続してアナログ入出力の実験を行うときの本システムの電源はバッテリから供給し，絶縁することを推奨する．ここでは，単3形ニッケル水素蓄電池を4本を入れた電池ボックスを拡張基板(SB)のACアダプタ・ジャックに接続した

付属基板MBに搭載されたH8SX/1655マイコン内蔵のA-D変換器とD-A変換器を使ってアナログ信号を入出力します．**写真1**に実験のようすを示します．

実験の準備

[**手順1**]統合開発環境Eclipseに，プロジェクトPROG_08_ADDAをインポートしてください．場所は，

C:¥CQ¥H8SX_1655¥software¥workspace¥PROG_08_ADDA

です．プロジェクトをクリーンアップして再ビルドしてください．

[**手順2**]拡張基板SBに付属基板MBを載せて，MBとパソコンをUSBケーブルで接続します．MBにプログラムをダウンロードします．MBのジャンパ・ピンJ_1をショートして，SB上のリセット・スイッチを押します．

ビルドしたプログラムのバイナリ・ファイル，

C:¥CQ¥H8SX_1655¥software¥workspace¥PROG_08_ADDA¥Debug¥PROG.mot

をFDTを使ってH8SX/1655にダウンロードしてください．

[**手順3**]MBのジャンパ・ピンJ_1をオープンにして，SB上のリセット・ボタンを押すとプログラムが走り出し，LCDモジュールに**写真1**のように表示されます．

実験結果

アナログ入力と出力はそれぞれ2チャネルあります拡張基板SBのステレオ・ジャックCN_{12}とCN_{13}

第17章 タッチ・パネルで操作できるボタンを作る実験

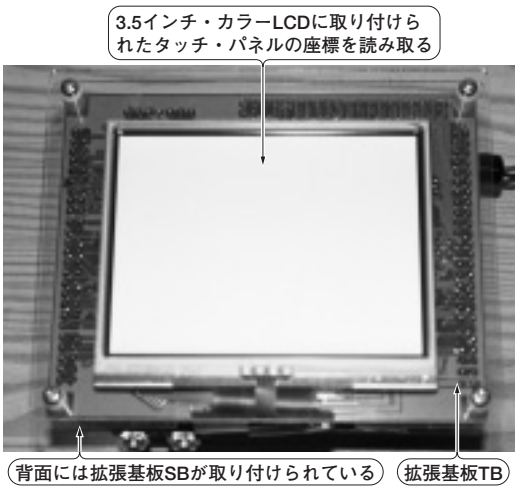
—— マイコン内蔵 A-D を使ってタッチ座標を検出する

本章では，TFT カラー LCD パネルに付属する抵抗膜式タッチ・パネルの座標読み取り方式を学びます．応用として LCD 画面上にタッチ・ボタンを実装する簡単なアプリケーション・プログラムを紹介します．

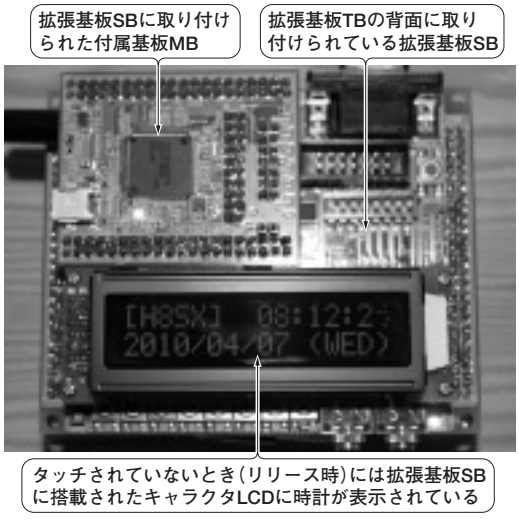
写真 1 に，付属基板 MB と拡張基板 TB，SB を使ったタッチ・パネルのタッチ座標を読み取る実験のようすを示します．

実験の準備

［**手順 1**］統合開発環境 Eclipse に，プロジェクト「PROG_10_TOUCH」をインポートします．場所は，



（**a**）実験ハードウェアの外観（LCDパネル側）



（**b**）実験ハードウェアの外観［（**a**）の裏側］

（**c**）タッチ時のキャラクタLCDの表示

写真 1　拡張基板 SB（第 8 章）と TB（第 9 章）を使ってタッチ座標の読み取り実験を行う
この実験は MB，SB，TB をすべて使用する．（a）は TB 裏側の LCD パネルである．このプログラムでは何も表示しない．（b）はタッチ・パネルに何もタッチしていないリリース状態の SB のキャラクタ LCD を示しており，RTC による時計を表示している．（c）はタッチ・パネルにタッチした状態のキャラクタ LCD の表示である．1 行目はタッチ位置の補正済み座標であり，2 行目は補正前の生データを示している