3 マイコンボード・開発環境の説明

3.1 マイコンボード MB-H8A

今回はルネサスエレクトロニクス株式会社 (Renesas Electronics Corporation) の CPU である、H8/3694 グループを利用することにします。プログラムの基本を勉強する間は、この CPU を搭載したサンハヤト株式会社の製品 MB-H8A を使います。MB-H8A の使用マイコンは H8/3694 グループの HD64F3694FX です。

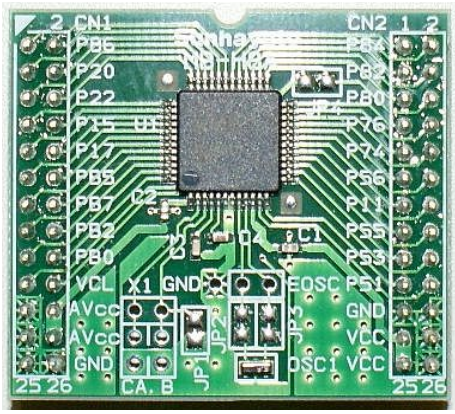

図 3.1　マイコンボード MB-H8A

表 3.1 に本テキストで使用する機能を中心に、H8/3694 グループの特徴を記述しておきます。

周辺機能については、テキスト中で解説しています。現時点で分からなくてもかまいません。

詳細は、「H8/3694 グループ ハードウェアマニュアル」を確認してください。

表 3.1　H8/3694 グループの特徴

CPU	16 ビット高速 H8/300H
内臓 ROM	32KB
内臓 RAM	2KB
クロック周波数	20MHz
I/O ポート	汎用入出力ポート 29 本、汎用入力ポート 8 本
タイマ A	時計用タイムベース機能を内蔵した 8 ビットタイマ
タイマ V	8 ビットタイマ
タイマ W	16 ビットタイマ
シリアルコミュニケーション インターフェース 3(SCI 3)	クロック同期／調歩同期モード
I2C バスインターフェース (IIC2)	
A/D コンバータ	10 ビット 8 チャンネル
電源電圧	DC4.0-5.5V

H8/3694 グループのアドレス空間はプログラム領域とデータ領域合わせて 64K バイトです。メモリマップを図 3.2 に示します。

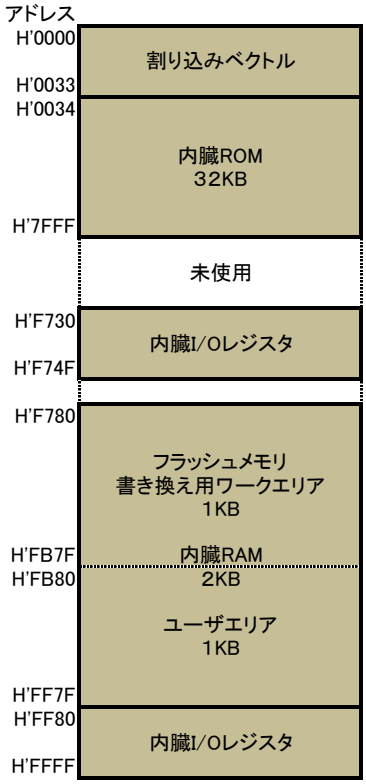

図 3.2　H8/3694 グループのメモリマップ

3.2 書込み・拡張 I/O ボード MB-RS10

マイコンボード MB-H8A を動作させるには、電源回路やプログラム書き込み用のシリアル回路などが必要になります。このための MB-H8A 用書込み・拡張 I/O ボード MB-RS10 がサンハヤト株式会社から発売されていますので、これも利用することにします (図 3.3)。

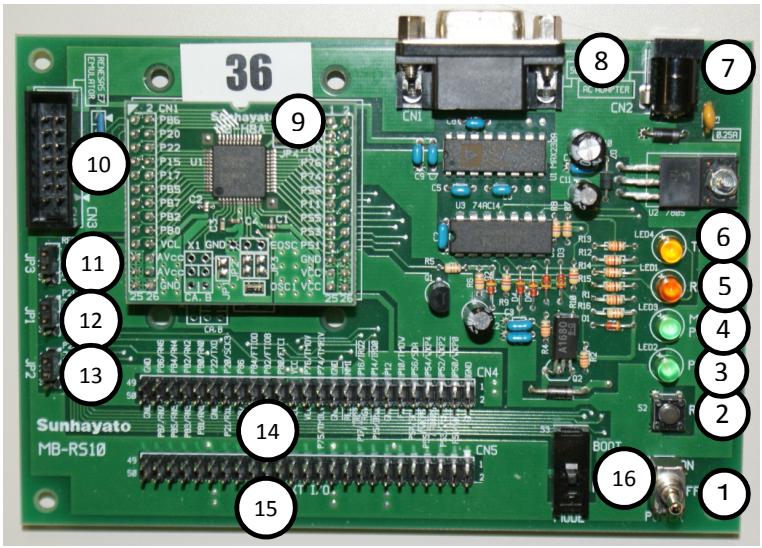

図 3.3　書込み・拡張 I/O ボード MB-RS10

各部の機能は以下の通りです。

① 電源スイッチ (POWER) : つまみを「ON」の側 (図上側) に倒すと電源が入ります。

② リセットスイッチ (RESET) : 押すとマイコンがリセットされます。

③ 電源ランプ (POWER) : 電源が入ると点灯します。

④ モジュール電源ランプ (MODULE POWER) : マイコンボードが正しく差し込まれていれば、電源が入った約 1 秒後に点灯します。電源スイッチを入れても点灯しない時には、マイコンボードの接続を確認してください。

⑤ データ受信ランプ (RX DATA) : パソコンからデータを受信しているときに点灯します。

⑥ データ送信ランプ (TX DATA) : マイコンボードからデータを送信しているときに点灯します。

⑦ DC ジャック (CN2) : AC アダプタを差し込みます。

⑧ シリアル通信コネクタ (CN1) : シリアル通信ケーブル (ストレート) のオス側を差し込みます。ケーブルのメス側はパソコンのシリアルポートに接続してください。

⑨ マイコンボード MB-H8A

⑩ E7 接続コネクタ (CN3) : ルネサスエレクトロニクス社製エミュレータ E7 を接続することができます。

⑪ リセットジャンパ (JP3) : リセット回路と拡張 I/O コネクタの $\overline{RES}$ 端子 (CN4-20,CN5-20) を接続する際には 2,3 ピンを接続します。本テキストでは、1,2 ピンを接続します。

⑫ シリアル通信ジャンパ (P21/RxD 用 JP1) : 1,2 ピンを接続することによって、レベル変換 IC を通してシリアル通信コネクタ (CN1) に接続されます (パソコンからデータを受信)。2,3 ピンを接続すると、拡張 I/O コネクタの P21/RxD 端子 (CN4-38,CN5-38) に接続されます。この際にはレベル変換は行われません。本テキストでは、1,2 ピンを接続します。

⑬ シリアル通信ジャンパ (P22/TxD 用 JP2) : 1,2 ピンを接続することによって、レベル変換 IC を通してシリアル通信コネクタ (CN1) に接続されます (マイコンボードからデータを送信)。2,3 ピンを接続すると、拡張 I/O コネクタの P22/TxD 端子 (CN4-39,CN5-39) に接続されます。この際にはレベル変換は行われません。本テキストでは、1,2 ピンを接続します。

⑭ 信号チェック用ピン (CN4) : マイコンボードの各信号が接続されています。

⑮ 拡張 I/O コネクタ (CN5) : マイコンボードの各信号が接続されています。各ピンの割り付けは信号チェック用ピン (CN4) と同じです。

⑯ モード切り替えスイッチ (MODE) : マイコンボードの動作モードを切り替えます。BOOT(プログラム書き込み) モードにするには、つまみを「BOOT」のほうにスライドさせます。RUN(プログラム実行) モードにするには、つまみを「RUN」のほうにスライドさせます。BOOT モードにすると、シリアル通信はレベル変換 IC を通して接続されます。

表 3.2 に信号チェック用ピン (CN4) および拡張 I/O コネクタ (CN5) の端子割付表を示しておきます。MB-RS10 の信号チェック用ピン (CN4) に記述されていますので確認してください。

表 3.2　MB-RS10 信号チェック用ピン (CN4) および拡張 I/O コネクタ (CN5) 端子割付表

番号	信号名	備考	番号	信号名	備考
1	GND		2	GND	
3	P50/$\overline{WKP0}$		4	P51/$\overline{WKP1}$	
5	P52/$\overline{WKP2}$		6	P53/$\overline{WKP3}$	
7	P54/$\overline{WKP4}$		8	P55/$\overline{WKP5}$/$\overline{ADTRG}$	
9	P56/SDA		10	P57/SCL	
11	P10/TMOW		12	P11	
13	P12		14	GND	
15	P14/$\overline{IRQ0}$		16	P15/$\overline{IRQ1}$	
17	P16/$\overline{IRQ2}$		18	P17/$\overline{IRQ3}$/TRGV	
19	$\overline{NMI}$		20	$\overline{RES}$	JP3(2,3)
21	GND		22	GND	
23	P74/TMRIV		24	P75/TMICV	
25	P76/TMOV		26	VCC	
27	VCC		28	VCC	
29	P80/FTCI		30	P81/FTIOA	
31	P82/FTIOB		32	P83/FTIOC	
33	P84/FTIOD		34	P85	
35	P86		36	P87	
37	P20/SCK3		38	P21/RXD	JP1(2,3)
39	P22/TXD	JP2(2,3)	40	GND	
41	PB0/AN0		42	PB1/AN1	
43	PB2/AN2		44	PB3/AN3	
45	PB4/AN4		46	PB5/AN5	
47	PB6/AN6		48	PB7/AN7	
49	GND		50	GND	

回路図はファイル「MB-RS10Schema.pdf」を確認してください。

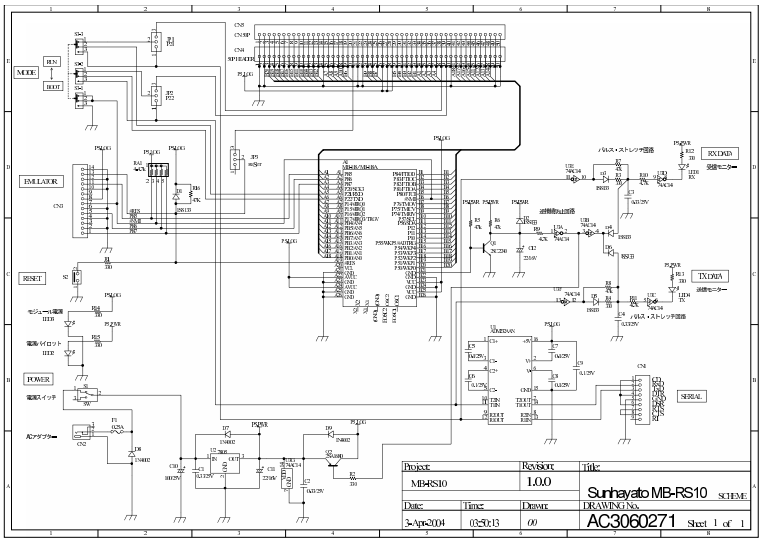

図 3.4　MB-RS10 の回路図 (MB-RS10Schema.pdf)

この書込み・拡張 I/O ボード MB-RS10 には制御するべき周辺機器が何もついていませんので、ブレッドボードを用いて必要な外部回路を構成します。構成した回路は、拡張 I/O コネクタと接続することになります。

ブレッドボードは部品やリード線を挿すだけで回路が組み立てられる回路基板です。その概観を図 3.5 に、図 3.6 に内部接続の様子を示します。

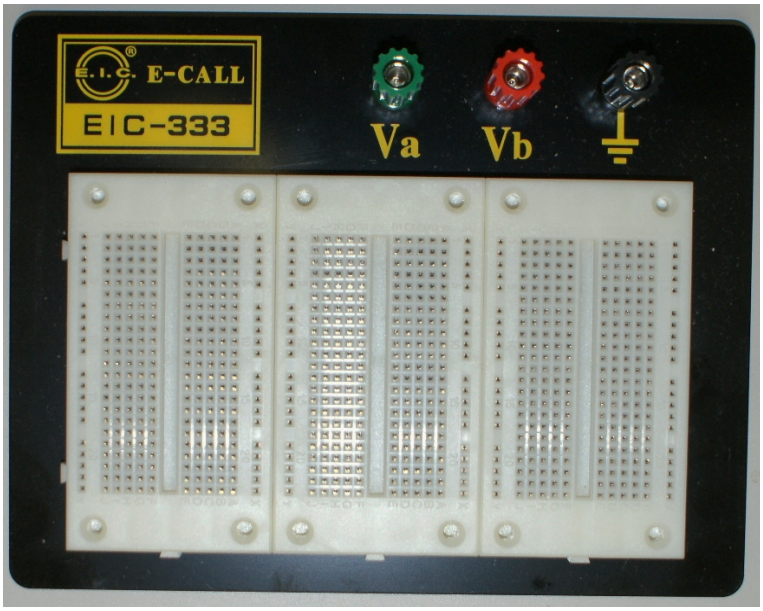

図 3.5　ブレッドボードの概観

ブレッドボードは、図 3.6 の右図のように、両端の縦に並んだ穴がつながっています。この部分は電源

(VCC) やグランド (GND) に利用します。中に並んだ穴は、中央から右の横一列と左の横一列がそれぞれつながっています。この部分に電子部品を配置します。

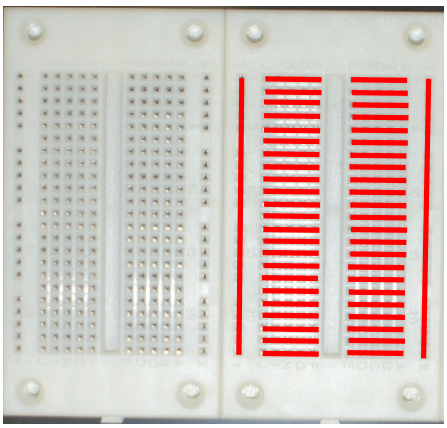

図 3.6　ブレッドボードの内部接続

3.3　マイコンプログラムの開発

パソコン上で動かすプログラムを開発する場合には、プログラムを書き、実行ファイルを作り、実行するという作業はすべてパソコン上で行います。

しかし、組込みマイコンのプログラムを開発する場合にはそうはいきません。組込みシステムにはキーボードやディスプレイが装備されていないことが普通です。その上、実行ファイルを作るための環境を持っているシステムも極めてまれです。そのため、パソコン上でプログラムを開発し、実行ファイルを作ります。この実行ファイルを組込みシステムに転送して、組込みシステム上で実行することになります。

このように、プログラムの開発環境と実行環境が異なるものをクロス開発環境といいます。組込みシステムのプログラム開発は、クロス開発環境で行われるわけです。

プログラムを開発するパソコン側をホスト、実行するマイコン側をターゲットと呼びます。

以下では、クロス開発環境でプログラムの開発をする方法を説明します。ここで利用する開発環境はルネサス エレクトロニクス株式会社の HEW です。また、ターゲットへの書き込みには同じルネサス エレクトロニクス株式会社の FDT を利用します。

3.4　HEW の利用 (実行ファイルの作り方)

統合開発環境 HEW(High-performance Embedded Workshop) は、ルネサス エレクトロニクス株式会社（Renesas Electronics Corporation）がマイコンの開発ツールとして販売しているソフトです。販売しているバージョンの他に、インターネットからダウンロードできる、無償評価版もあります。

ここでは、HEW を使ってソースファイル (プログラム) を入力し、実行ファイルを作るまでの手順を説明します。

3.4.1 ワークスペースとプロジェクト

ここでは HEW による実行ファイルの作成方法を説明します。

まずは、ワークスペースとプロジェクトを作りましょう。

プロジェクトというのは、関連するファイルをひとまとまりにしておく箱のようなものです。一つの実行ファイルを作るのには、HEW が自動で生成するファイルなど、複数のファイルが必要になってきます。これらをプロジェクトというものに整理しておくと分かりやすくなるわけです。

HEW ではプロジェクトをひとまとまりにしておくために、さらにワークスペースという箱のようなものを用意しています。これには、たとえば H8_3694F のプロジェクト全体 (LED の実行ファイルを作るためのプロジェクトとか、スイッチの実行ファイルを作るためのプロジェクトなど) のような、同じマイコンのプロジェクトを入れておくと便利です (図 3.7)。

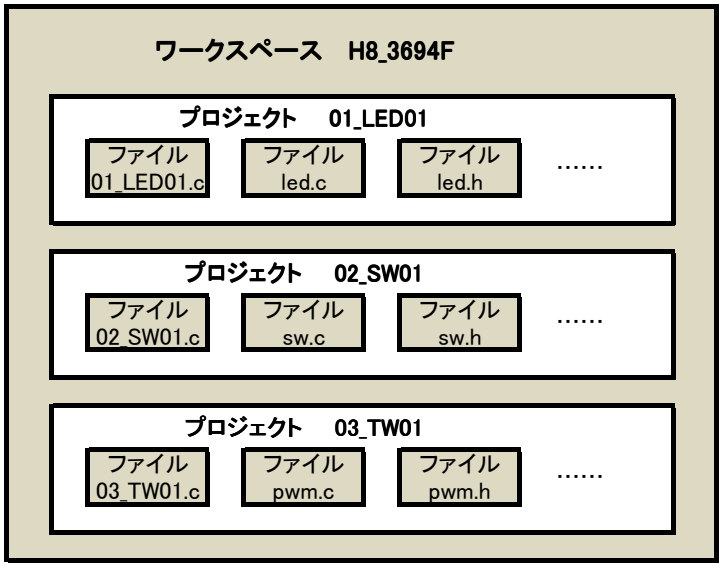

図 3.7　ワークスペース、プロジェクト、ファイルの関係

このようにしておくと、LED のプログラムを編集した後に、スイッチのプログラムを編集しようと思ったら、アクティブなプロジェクトを切り替えるだけですみ、プロジェクトを終了したり起動したりという手間が省けるのです。

ここでは、ワークスペース名を「H8_3694F」とし、プロジェクト名は適宜つけることにしていきましょう。プロジェクトは名前でソートされますので、01LED、02SW など名前の先頭を数字にすると良いようです。

3.4.2 ワークスペースとプロジェクトの作成 (ワークスペースを作る)

それでは、新規にワークスペースを作ってみましょう。

まずは HEW を起動します。「スタート」-「すべてのプログラム」-「Renesas」-「High-performance Embeded Workshop」-「High-performance Embeded Workshop」の順でクリックしてください。

HEW が起動すると同時に、図 3.8 のような、「ようこそ！」というダイアログボックスが表示されます。ここでは新たにワークスペースを作りたいので、「新規プロジェクトワークスペースの作成」にチェックをつけて、「OK」ボタンをクリックします。

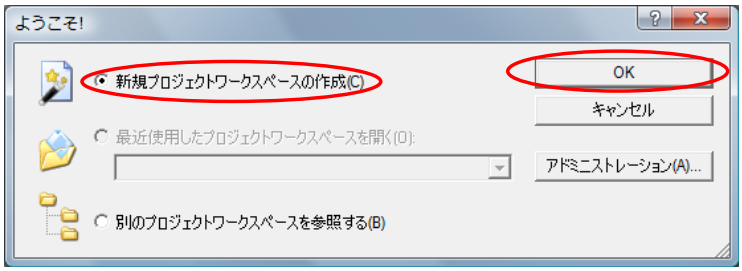

図 3.8　新規プロジェクトワークスペースの作成

次に、「新規プロジェクトワークスペース」というダイアログボックスが表示されますので、以下のように入力してください。

- ワークスペース名：H8_3694F
- プロジェクト名　：01_LED02

「ディレクトリ」は「参照…」ボタンをクリックして、Z ドライブを選択してください。選択が正しくできていれば、図 3.9 のように「Z:¥H8_3694F」と表示されます。「OK」ボタンをクリックします。

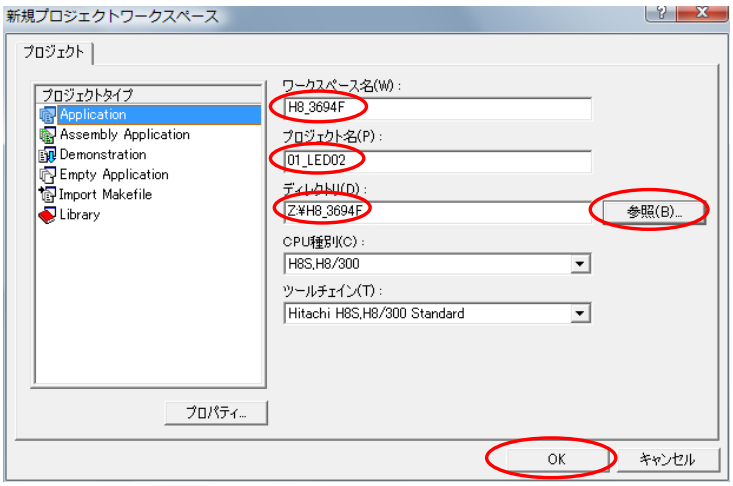

図 3.9　新規プロジェクトワークスペースの作成

「CPU」ダイアログボックスが表示されますので、以下のように選択してください (図 3.10)。

- CPU シリーズ : 300H
- CPU タイプ　 : 3694F

「次へ」ボタンをクリックします。

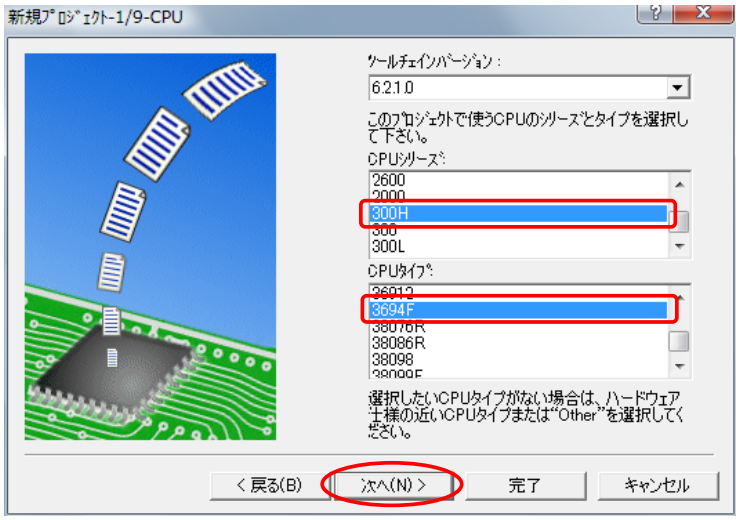

図 3.10　CPU の選択

次の画面 (図 3.11) は、グローバルオプションの指定です。ここは変更せずに「次へ」ボタンをクリックします。

図 3.11　オプションの選択

次の画面 (図 3.12) は、自動生成ファイルの指定です。「I/O レジスタ定義ファイル」にチェックが付いていることを確認します。「次へ」ボタンをクリックしましょう。

図 3.12　生成ファイルの選択

次の画面 (図 3.13) は、標準ライブラリの指定です。このテキストでは、移植性を考慮して標準ライブラリをできるだけ使わずにプログラムをします。チェックを外してもかまいませんが、そのままで「次へ」ボタンをクリックしましょう。

図 3.13　標準ライブラリの選択

次の画面 (図 3.14) は、スタック領域の設定です。スタック領域は正確に見積もる必要があるのですが、このテキストの範囲ではそこまでやりません。そのままで「次へ」ボタンをクリックしましょう。

図 3.14　スタック領域の設定

次の画面 (図 3.15) は、ベクタの設定です。割込みの際に制御がどこに移るかの記述するファイルを生成します。そのままで「次へ」ボタンをクリックしましょう。

図 3.15　ベクタの設定

次の画面 (図 3.16) は、デバッガの設定です。デバッガを利用する場合にチェックをつけます。今回は利用しないので、そのままで「次へ」ボタンをクリックしましょう。

図 3.16　デバッガの設定

次の画面 (図 3.17) は、プロジェクトの概要の確認です。内容を確認して、そのままで「次へ」ボタンをクリックしましょう。

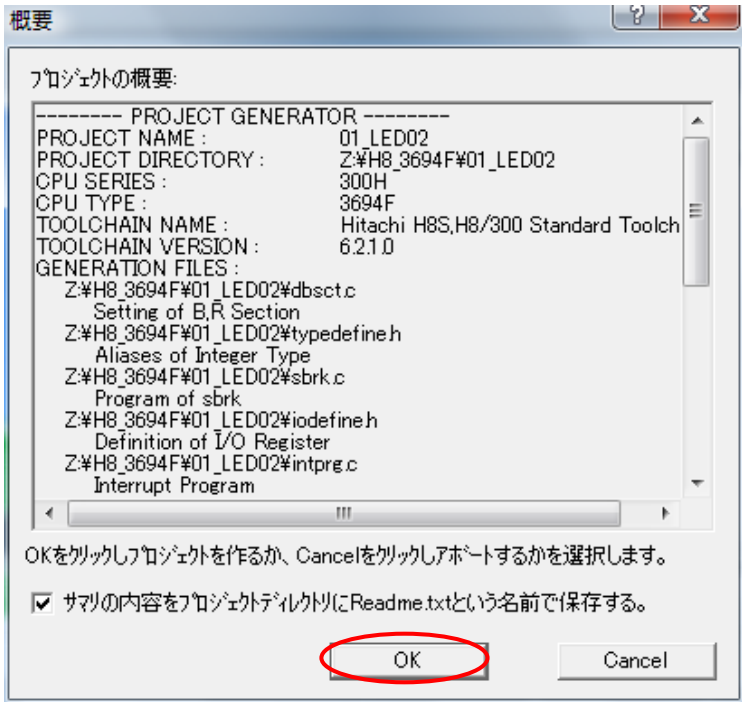

図 3.17　プロジェクトの概要

以上の操作を行うと図 3.18 のような、表示になります。

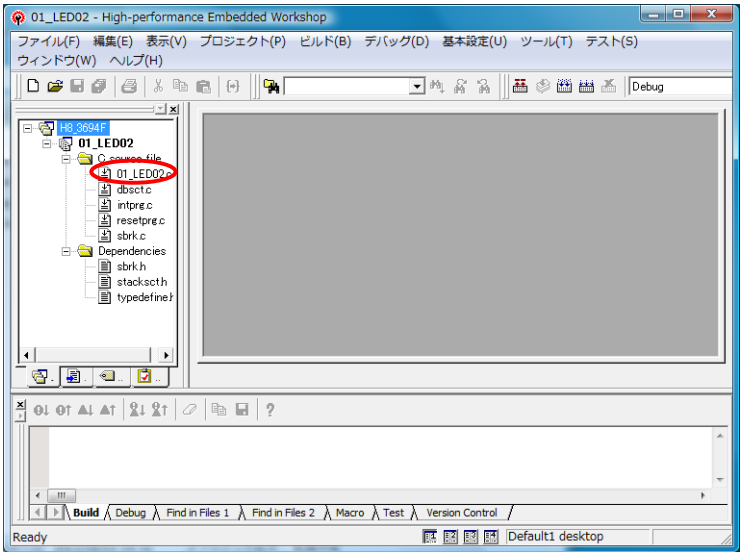

図 3.18　プロジェクトの生成

左側のウィンドウ (ワークスペースウィンドウ) 内にプロジェクト名と同じ「01_LED02.c」というファイルがあることが分かります。main 関数はこのようなプロジェクト名と同じ名前の C 言語のファイルに

記述します。

ファイルの中身を表示するには、ファイル名をダブルクリックします。ワークスペースウィンドウ内の「01_LED02.c」をダブルクリックしてみましょう。

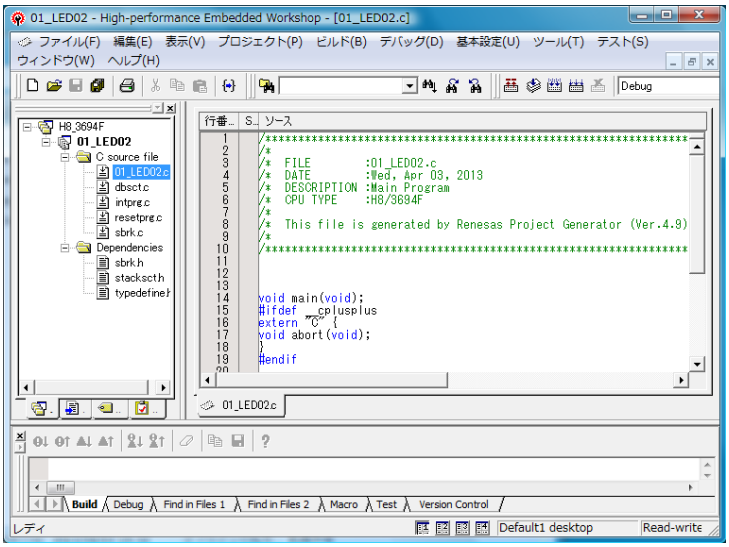

図 3.19　ファイルの表示

図 3.19 のように、右側のウィンドウ (エディタウィンドウ) にファイルの中身が表示されます。

01_LED02.c の内容は以下のようになっています。

01_LED02.c(変更前)

```
 1  /***********************************************************************/
 2  /*                                                                     */
 3  /*  FILE        :01_LED02.c                                            */
 4  /*  DATE        :Wed, Apr 03, 2013                                     */
 5  /*  DESCRIPTION :Main Program                                          */
 6  /*  CPU TYPE    :H8/3694F                                              */
 7  /*                                                                     */
 8  /*  This file is generated by Renesas Project Generator (Ver.4.9).     */
 9  /*                                                                     */
10  /***********************************************************************/
11
12
13  void main(void);
14  #ifdef __cplusplus
15  extern "C" {
16  void abort(void);
17  }
18  #endif
19
20  void main(void)
21  {
22
23  }
24
25  #ifdef __cplusplus
26  void abort(void)
27  {
28
29  }
30  #endif
```

End Of List

これは HEW が自動的に作るファイルですが、C 言語でプログラムする場合には、なくても特に問題はないようです。このテキストでは、以下のように、適宜コメントを書き換え、main 関数以外を削除して掲載します。

01_LED02.c(変更後)

```
 1  /************************************************************************/
 2  /*                                                                      */
 3  /*  FILE        :01_LED02.c                                             */
 4  /*  DESCRIPTION :LED1 の点灯 (キャストの利用)                           */
 5  /*  CPU TYPE    :H8/3694F                                               */
 6  /*                                                                      */
 7  /*  LED0 P85                                                            */
 8  /*  LED1 P86                                                            */
 9  /*  LED2 P87                                                            */
10  /*                                                                      */
11  /*  This file is generated by Renesas Project Generator (Ver.4.9).     */
12  /*                                                                      */
13  /************************************************************************/
14
15  void main(void)
16  {
17
18  }
```

End Of List

この作業は、必ずしも必要ありませんが、本テキストでは説明の都合上、このように不用な部分を削除したものを用いて説明していきます。

3.4.3 ソースファイル (プログラム) の作成

実行ファイルを作るためには、ソースファイルを作らなければなりません。

ここでは具体例として、LED を点滅するプログラムを記述してみましょう。プログラムの詳細については、P. 98 の「LED1 の点灯 (キャストの利用)」で説明します。

01_LED02.c

```
 1  /************************************************************************/
 2  /*                                                                      */
 3  /*  FILE        :01_LED02.c                                             */
 4  /*  DESCRIPTION :LED1 の点灯 (キャストの利用)                           */
 5  /*  CPU TYPE    :H8/3694F                                               */
 6  /*                                                                      */
 7  /*  LED0 P85                                                            */
 8  /*  LED1 P86                                                            */
 9  /*  LED2 P87                                                            */
10  /*                                                                      */
11  /*  This file is generated by Renesas Project Generator (Ver.4.9).     */
12  /*                                                                      */
13  /************************************************************************/
14
15  void main(void)
16  {
17    /*********************************************
18      LED 接続端子の初期化
19    *********************************************/
20    (*((volatile unsigned char *)0xFFEB)) = 0xE0;
21
22    /*********************************************
23      LED1 を点灯他は消灯
24    *********************************************/
25    (*((volatile unsigned char *)0xFFDB)) = (~0x40);
26
27    /*********************************************
28      無限ループ
29    *********************************************/
30    while(1){
31      ;
32    }
33  }
```

End Of List

入力が終わったら、「ファイルの保存」ボタン もしくは、「すべて保存」ボタン をクリックしてファイルを保存します。このとき、タイトルバーに表示されているファイル名が、「01_LED02.c*」から「01_LED02.c」に変わることを確認してください (図 3.20)。

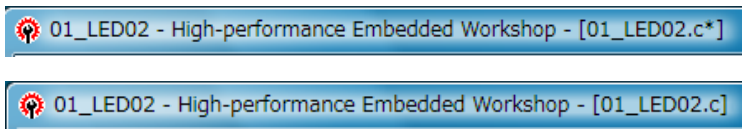

図 3.20　タイトルバーの表示 保存前 (上)、保存後 (下)

3.4.4 プロジェクトをビルドする

実行ファイルを作るためには、ビルドを行います。

前回のビルド後に変更のあったファイルだけをビルドする場合には、「ビルド」ボタン をクリックし、すべてをビルドしなおすときには、「すべてをビルド」ボタン をクリックします。

ビルドが終わると下のウィンドウ (アウトプットウィンドウ) に結果が表示されます (図 3.21)。

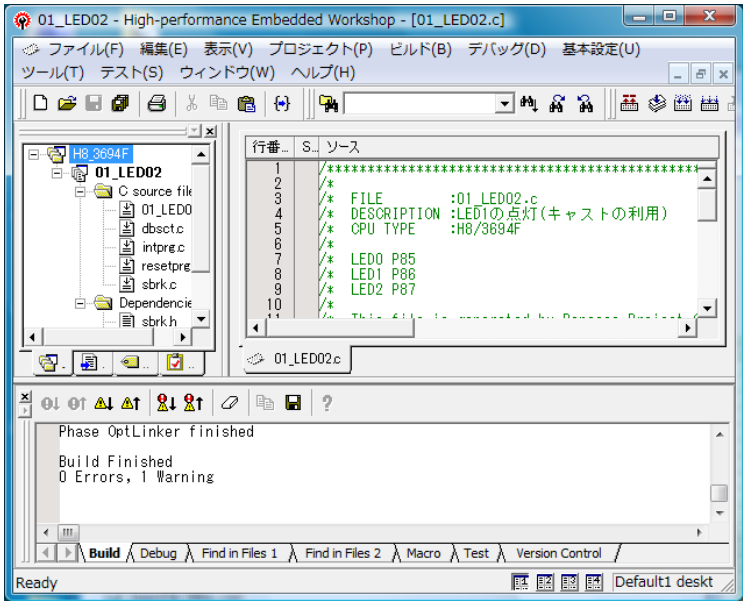

図 3.21　ビルドの結果

ここでは Warning が一つ出ていますが、気にしないことにします (Constant 型のデータがないので、Warning が出ています。C セクションを削れば消えます)。

3.4.5 以前に作成したワークスペースを開く (2 回目以降の HEW の起動)

上記の設定をして、2 回目以降に HEW を起動した場合、「最近使用したプロジェクトワークスペースを開く」という項目にチェックがつき、前回起動したワークスペースのパスが表示されています (図 3.22)。

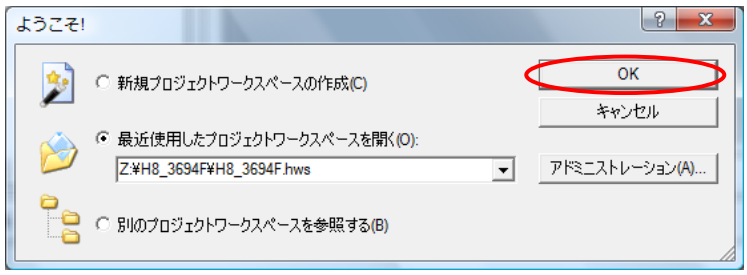

図 3.22　2 回目以降の起動

そのまま「OK」ボタンをクリックすると、前回のワークスペースを開くことができます。