

OAKS8

クイックツアー

安全設計に関するお願い

弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご注意ください。

本資料ご利用に際しての留意事項

本資料は、お客様が用途に応じた適切な製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてオークス電子および情報を提供いただいた各社が所有する知的財産権その他の権利の実施、使用を許諾するものではありません。

本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、オークス電子は責任を負いません。

本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、オークス電子は特性改良などにより予告なしに変更することがあります。

本資料に記載の図、表に示す技術的な内容、及びプログラム、アルゴリズムを流用する場合、お客様の責任において実施してください。また、組み込んだプログラム、アルゴリズム単体で評価するだけでなく、システム全体で十分に評価してください。オークス電子は、一切責任を負いません。

本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、オークス電子へご照会ください。

本資料の転載、複製については、文書によるオークス電子の事前の承諾が必要です。

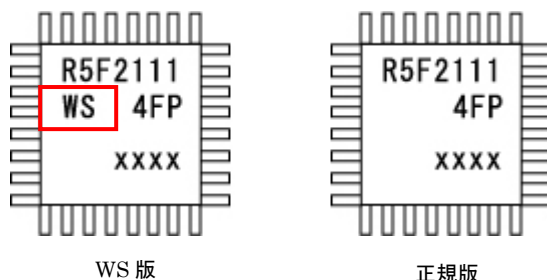
本資料に関し詳細についてのお問い合わせ、その他お気づきの点がございましたらオークス電子までご照会ください。

Microsoft, MS 及びMS-DOS は、米国Microsoft Corporation の登録商標です。
 Windows95, Windows98 は、米国Microsoft Corporation の商標です。
 IBM 及びPC/AT は、米国International Business Machines Corporation の登録商標です。
 Pentium は、米国Intel Corporation の商標です。
 Adobe, Acrobat は、Adobe Systems Incorporated(アドビシステムズ社)の商標です。

ーはじめにー

OAKS8に搭載されておりますマイクロコンピュータ R5F21114FP には、WS (ワークサンプル) 版 (初期ロットのみ) と正規版があります。WS版と正規版は、機能的には同じものです。相違点は以下のとおりです。

- WS版は、チップ表面に“WS”と印字があります。正規版には“WS”の印字がありません。



- 半導体工場での出荷時の品質テストレベルが違います。WS版を量産品に組み込む事はできません。
- 内部ブートエリアにあらかじめ書き込まれているブートプログラムのバージョンが違います。
 WS版→Ver 0.9 正規版→Ver 1.0

ブートプログラムのバージョンが違う事で、リモートデバッグKD30の起動時の動作、メモリマップ、設定するIDコードに違いがあります。

	WS版 (ブートプログラム Ver0.9)	正規版 (ブートプログラム Ver1.0)
KD30起動時の動作	現在のフラッシュメモリの内容を全て消去してから、モニタプログラムを書き込み、モニタプログラムへ実行を移します。	現在のフラッシュメモリに設定されているIDコードがすべて“00h”か“FFh”のときのみ、フラッシュメモリの内容を全て消去してから、モニタプログラムを書き込みモニタプログラムへ実行を移します。IDコードが“00h”か“FFh”以外の場合は、何もしません。 (ホストPCでは、モニタプログラム書き込みのプログラレスパーが止まった状態のままとなります)
KD30が使用する領域	RAM : 700H~7FFH (400H~6FFHまで使用可) ROM : C000H~0CBFFH (C000H~FFFFHまで使用可)	RAM : 制限なし ROM : C000H~0C7FFH (C800H~FFFFHまで使用可)
KD30起動の際に設定するIDコード	IDコードのID3, 4, 6に“00h”を設定します。ユーザプログラムをダウンロードしたあとは、IDコードは以下のようになります。 xx xx 00 00 xx 00 xx (xはユーザが設定した値) KD30起動後、ユーザプログラムをダウンロードしないで終了した場合は、以下のようになります。 FF FF 00 00 FF 00 FF	すべてのIDコードに“FFh”を設定します。KD30起動後は、ユーザプログラムをダウンロードしてもしなくてもIDコードはAll FFhになります。

本マニュアルでは、搭載されているマイクロコンピュータが正規版であることを前提に説明しております。お持ちのキットにWS版が搭載されている場合は、必ず「第12章 WS版における相違点」を参照してください。(サンプルプログラムの設定変更が必要です。)

一本マニュアル内での動作環境について

ソフトウェア (ツール、サンプルプログラム) は全てWindows2000の環境で動作させたものとしています。

目次

ーはじめにー	3
ー本マニュアル内での動作環境についてー	3
1. OAKS8-Ful IKit でのプログラム開発環境.....	7
1.1 プログラム開発フロー	7
2. クイックツアーのための準備	8
2.1 プログラム開発に必要なもの	8
2.2 テンプレートプログラムをコピーする.....	8
3. TM で新規プロジェクトをつくる.....	11
3.1 新規プロジェクトの作成	11
3.1.1 TM を起動する	11
3.1.2 プロジェクトの設定をする.....	12
3.1.3 リンクするすべてのファイルを登録する	16
3.2 プロジェクト作成時に設定されているオプション	17
3.3 KD30 でデバッグをする場合に設定が必要なオプション(-g オプション)	17
4. Peggy PAD でプログラムを書く.....	18
4.1 テンプレート「MAIN.C」の内容	18
4.2 Peggy PAD の表示設定をする.....	19
4.3 テンプレートにプログラムを書く.....	20
5. TM でビルドする.....	22
5.1 プロジェクトをビルドする.....	22
5.2 エラーが出たら	23
6. KD30 でプログラムを実行する	24
6.1 KD30 の起動からプログラムの実行まで	24
7. 実機でプログラムを実行する.....	26
8. Flash Starter でプログラムを書き込む	27

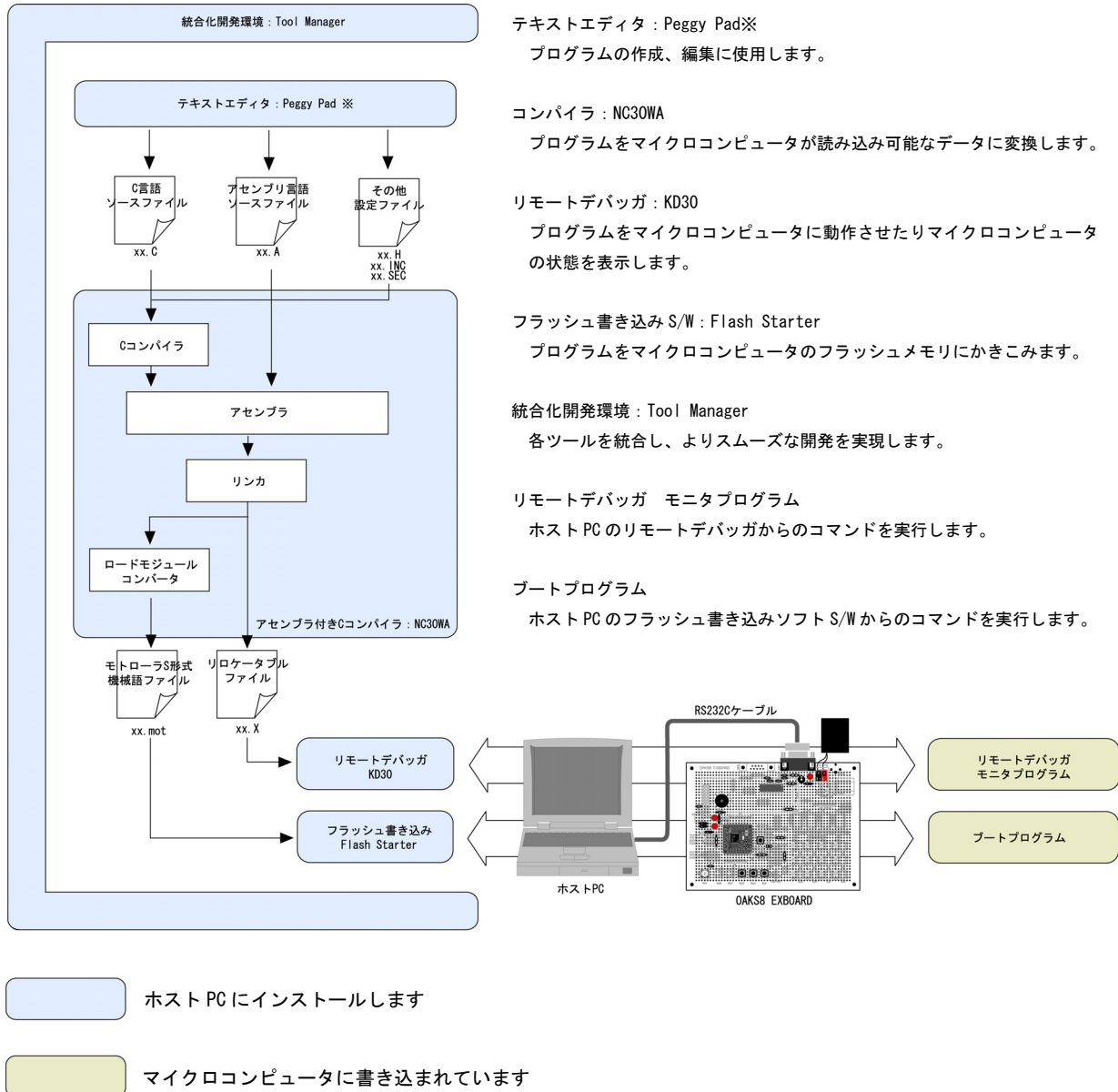
8.1 TM での設定を追加する.....	27
8.2 Flash Starter でプログラムを書き込む	29
9. KD30 とFlash Starter の動作について	31
9.1 KD30 について.....	31
8.1.1 KD30 について	31
9.1.2 KD30 起動時のマイクロコンピュータの動作モードについて.....	32
9.1.3 KD30 の動作について.....	33
9.1.4 KD30 終了後の実機動作について	34
9.1.5 他の OAKS シリーズの KD30 が既にインストールされている場合.....	34
9.2 Flash Starter について.....	35
9.2.1 Flash Starter の動作モードについて.....	35
9.2.2 Flash Starter の動作について	35
10. ID コード.....	36
10.1 ID コードについて	36
10.1.1 ID コードとは	36
10.1.2 ID コードの設定例	37
10.2 KD30 モニタプログラムが設定する ID コード(重要)	38
10.3 現在書き込まれている ID コードがわからなくなったら(重要)	39
10.3.1 機械語ファイルを参照する.....	39
10.4 付録のサンプルプログラムでの ID コード設定	39
11. 制限事項.....	40
11.1 メモリ制限.....	40
11.2 レジスタ操作に関する制限	41
11.3 周辺機能に関する制限	41
11.4 KD30 の機能に関する制限.....	42
11.3.1 サンプリングモードとフリーランモードについて	42
11.3.2 ストップモード、ウェイトモードに関する制限事項.....	42
11.3.3 監視タイマのリアルタイム性について	42
11.3.4 例外的なステップ実行について.....	43

12. WS版における相違点	44
12.1 WS版と正規版の主な相違点	44
12.2 ブートプログラム Ver0.9 の場合のメモリマップ	45
12.3 ブートプログラム Ver0.9 の場合に KD30 が設定する ID 番号	46
12.3.1 KD30 でユーザプログラムをダウンロードした場合設定される ID	46
12.4 ブートプログラムのバージョン確認方法	47
12.5 WS版でサンプルプログラムを動作させる場合の修正事項	48
12.5.1 プログラム開始アドレスの修正	48
12.5.2 スタックポインタ初期値の修正	48
13. 保証とサポート	49
13.1 保証について	49
13.2 技術サポートについて	49
改訂記録	50

1. OAKS8-FullKit でのプログラム開発環境

1.1 プログラム開発フロー

OAKS8-FullKit でのプログラムの開発フローを以下に示します。



※テキストエディタはアンカーシステムズ株式会社の「Peggy Pad」を紹介しておりますが、すでにお使いのものがあればインストールの必要はありません。

2. クイックツアーのための準備

2.1 プログラム開発に必要なもの

OAKS8-FullKitでのプログラムの開発には以下のものがが必要です。

- OAKS8-FullKitに付属のCD-ROM (OAKS8のラベル)
- OAKS8-EXBOARD (CPUボード、OAKS8-FullKitに付属の部品 (オプション用ジャンパを除く) を全て実装したもの)

以下のものは、別途ご用意ください。

- ホストパーソナルコンピュータ (IBM PC/ATシリーズおよびその互換機) → 各ツールがインストール済みのもの

CPU : Pentium II 233MHz 以上を推奨
OS : Microsoft Windows95/98/ME/NT/2000/XP
ブラウザ: Microsoft Internet Explorer 4.0以上 (統合化開発環境:TM 使用時に必要です)
メモリ : 128Mバイト以上を推奨
ポート : シリアルポート 1チャンネル
その他 : CD-ROMドライブ

- RS232Cケーブル: 9pinオスメス型ストレートケーブル (OAKS8-EXBOARDとホストPCとの接続に必要です)

- 電源用単三電池 3個 (マンガン又はアルカリ電池を推奨します)

2.2 テンプレートプログラムをコピーする

プログラムのテンプレートをホストPCにコピーします。

ホストPCのCD-ROMドライブに、OAKS8-FullKitのCD-ROMを挿入してください。挿入後にブラウザが自動起動し、HTMLの初期画面が表示されます。

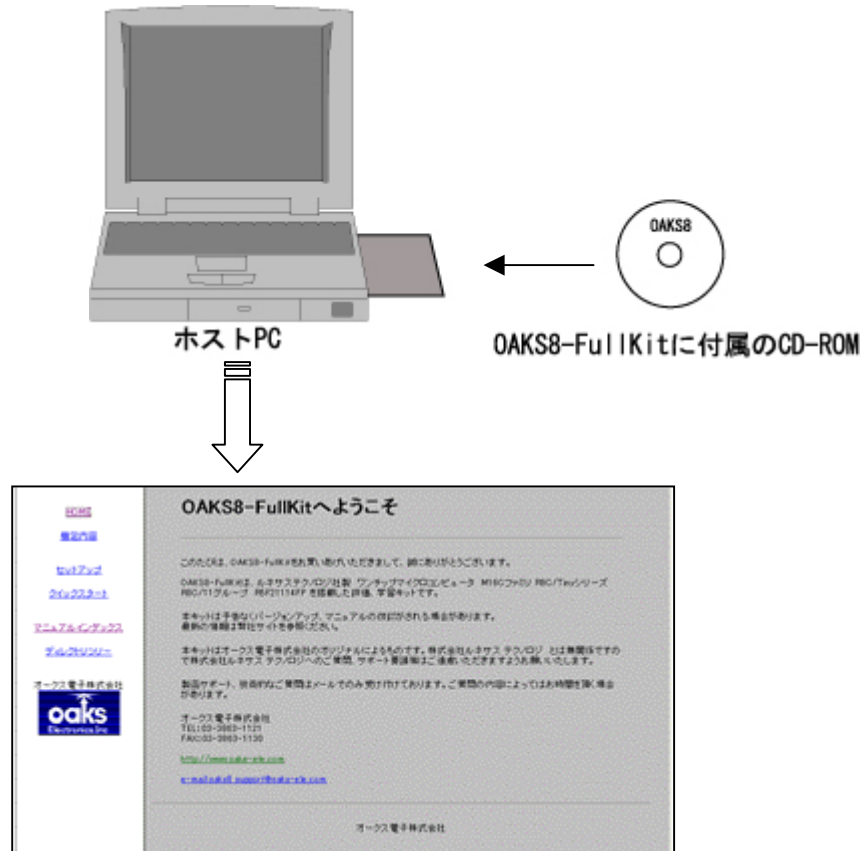


図2.1 OAKS8 CD-ROM初期画面

ホストPCの設定やお使いの環境によってはブラウザが自動起動しない場合があります。その場合はCD-ROMのファイルをエクスプローラで表示し、HTML初期画面ファイル“OAKS8.htm”をダブルクリックしてください。

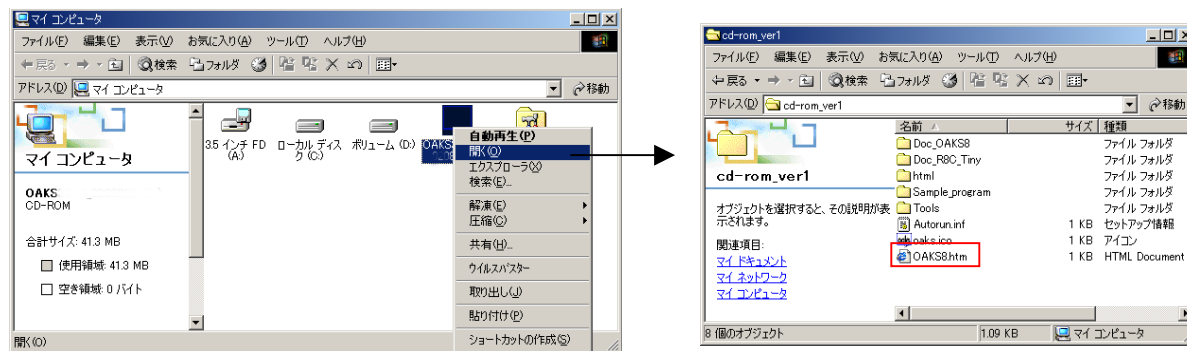


図2.2 “OAKS8.htm”を開く

左メニューより「セットアップ」をクリックし、STEP2の「サンプルプログラムのコピーバッチファイルを表示する」をクリックしてください。画面下のフレームに“COPY_SMP.BAT”が表示されます。“COPY_SMP.BAT”をダブルクリックするとMTOOLディレクトリ以下に“SAMPLE”というディレクトリを作成し、全てのサンプルプログラムをコピーします。(KD30、NC8、TMをC:\MTOOL以下にインストールした場合を想定しています。MTOOLディレクトリを作成しなかった場合は手動でコピーしてください。)

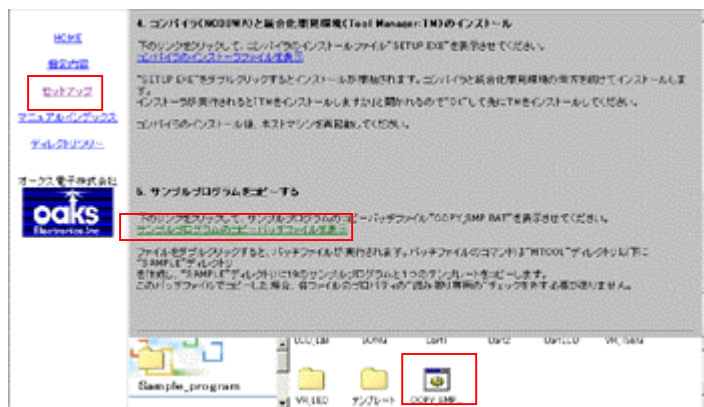


図2.3 サンプルプログラムコピーバッチファイルの表示

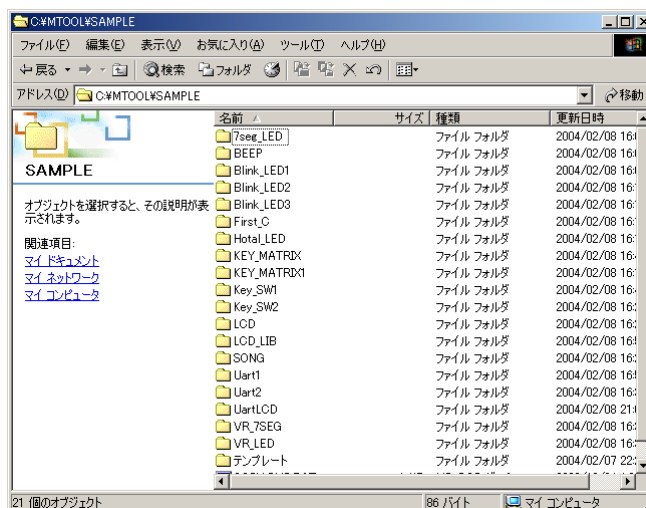


図2.4 コピーされたサンプルプログラム

任意のディレクトリにコピーする場合は、バッチファイルを使用しないで手でコピーしてください。その際、かならずすべてのファイルのプロパティの“読み取り専用”チェックを外してください。（バッチファイルでコピーした場合は必要ありません）
TMでは全角文字、スペースをふくむディレクトリ名（“マイ ドキュメント”等）は指定できませんのでご注意ください。

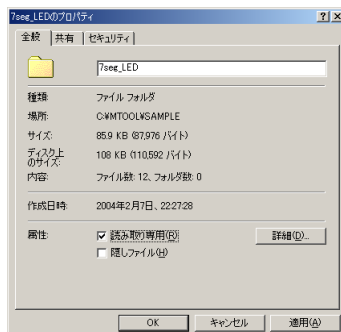


図2.5 フォルダのプロパティ

コピーしたサンプルプログラムの中の「テンプレート」を使ってプログラムを作成します。TMでは全角文字のディレクトリ名は使用できませんので、半角名の名前に変えます。任意の名前がかまいませんが、ここでは「FIRST」とします。

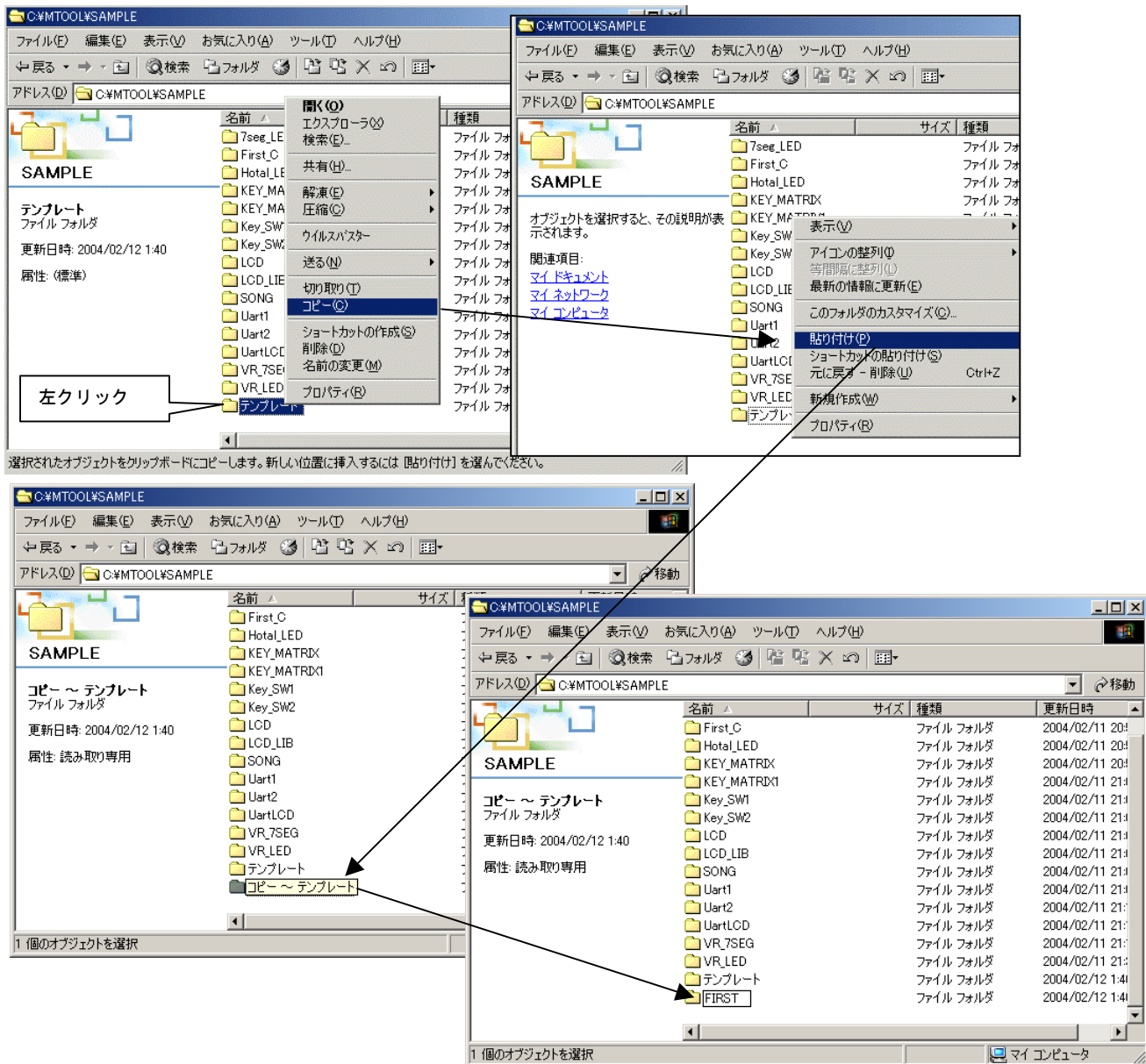


図2.5 テンプレートのリネーム

3. TMで新規プロジェクトをつくる

TM（統合化開発環境）はプログラム開発に必須のツールというわけではありませんが、ツール群の統合のみならず、コンパイル／アセンブルの手順作成（MAKEファイル）、オプションの付加なども非常に簡単に行えるものです。このツアーでは、TMでのプログラム開発を順を追って説明します。

3.1 新規プロジェクトの作成

「FIRST」ディレクトリ以下のプログラムのプロジェクトを作成します。プロジェクトとは、ソースファイル群やコンパイル／アセンブル手順、オプション情報などをひとまとめに呼ぶものです。基本的には、プロジェクトに関連するファイルは全て一つのディレクトリ内に収めます。（ディレクトリ内で複数のディレクトリに分かれていてもかまいません）

以下に「FIRST」フォルダ内の各ファイルの内容を示します。

ディレクトリ/ファイル名	内容	
C:\%MTOOL%\SAMPLE\FIRST	ncrt0.a30	OAKS8-FullKit用初期設定プログラム（アセンブラ言語）
	main.C	メインプログラム（C言語）
	sect30.inc	セクション定義ファイル（ncrt0.a30でインクルード）
	R8C11.H ※	ヘッダファイル（R8C/11グループのSFR定義、main.Cでインクルード）

※ヘッダファイルR8C11.Hはルネサス テクノロジー社製オリジナルのものではありません。

3.1.1 TMを起動する

Windowsの「スタート」メニューから [プログラム]－[RENEASAS-TOOLS]－[TM V3.20A]－[TM]をクリックしてください。プロジェクトバーが開きます。プログラム開発のための各ツールはプロジェクトバーより起動します。

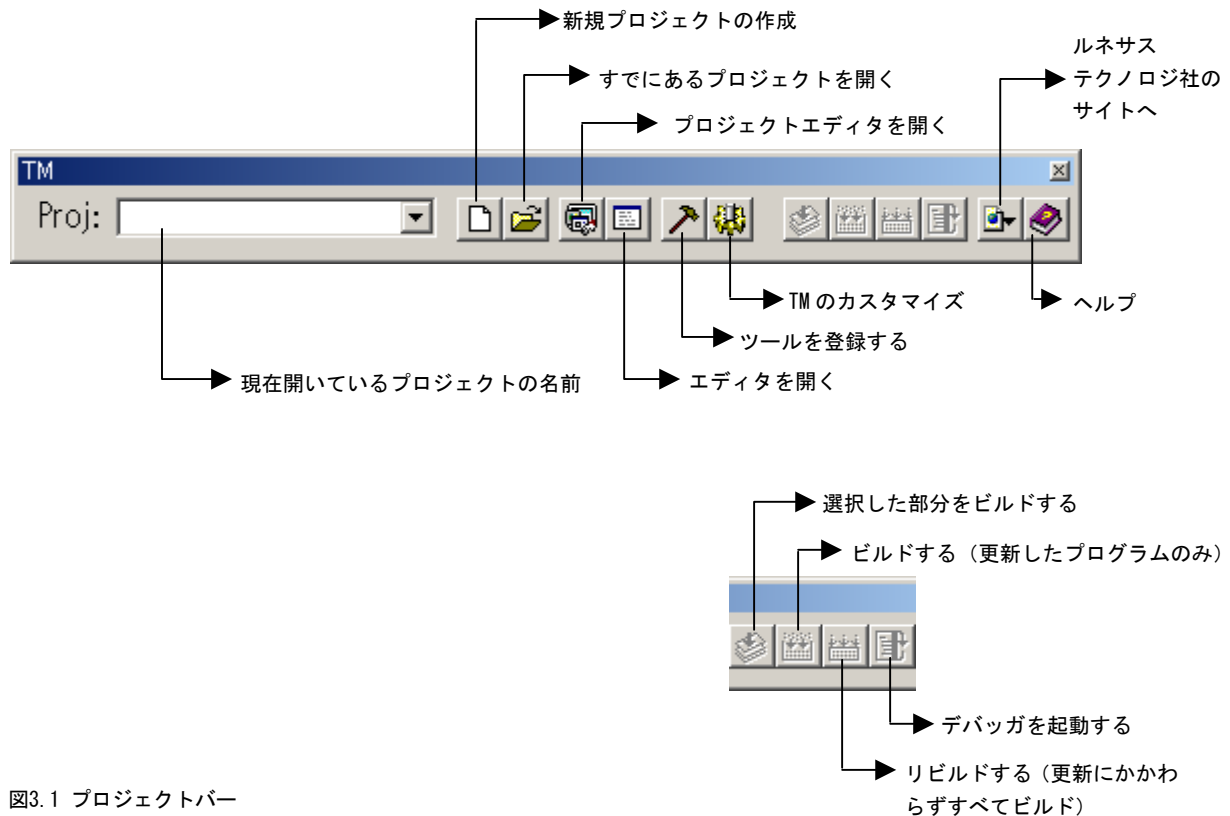


図3.1 プロジェクトバー

3.1.2 プロジェクトの設定をする

「New Project」ボタンをクリックし、ウィザードに従ってプロジェクトを作成します。

ウィザードの「ステップ1」では以下のように設定してください。

- ①ターゲットチップ：R8C/Tiny Series
- ②プロジェクト名：FIRSTとします。生成されるオブジェクトファイル名等に反映されます。
- ③ワーキングディレクトリ：C:\%MTOOL%\SAMPLE\FIRSTとします。ビルドによって生成されるファイルが格納されます。

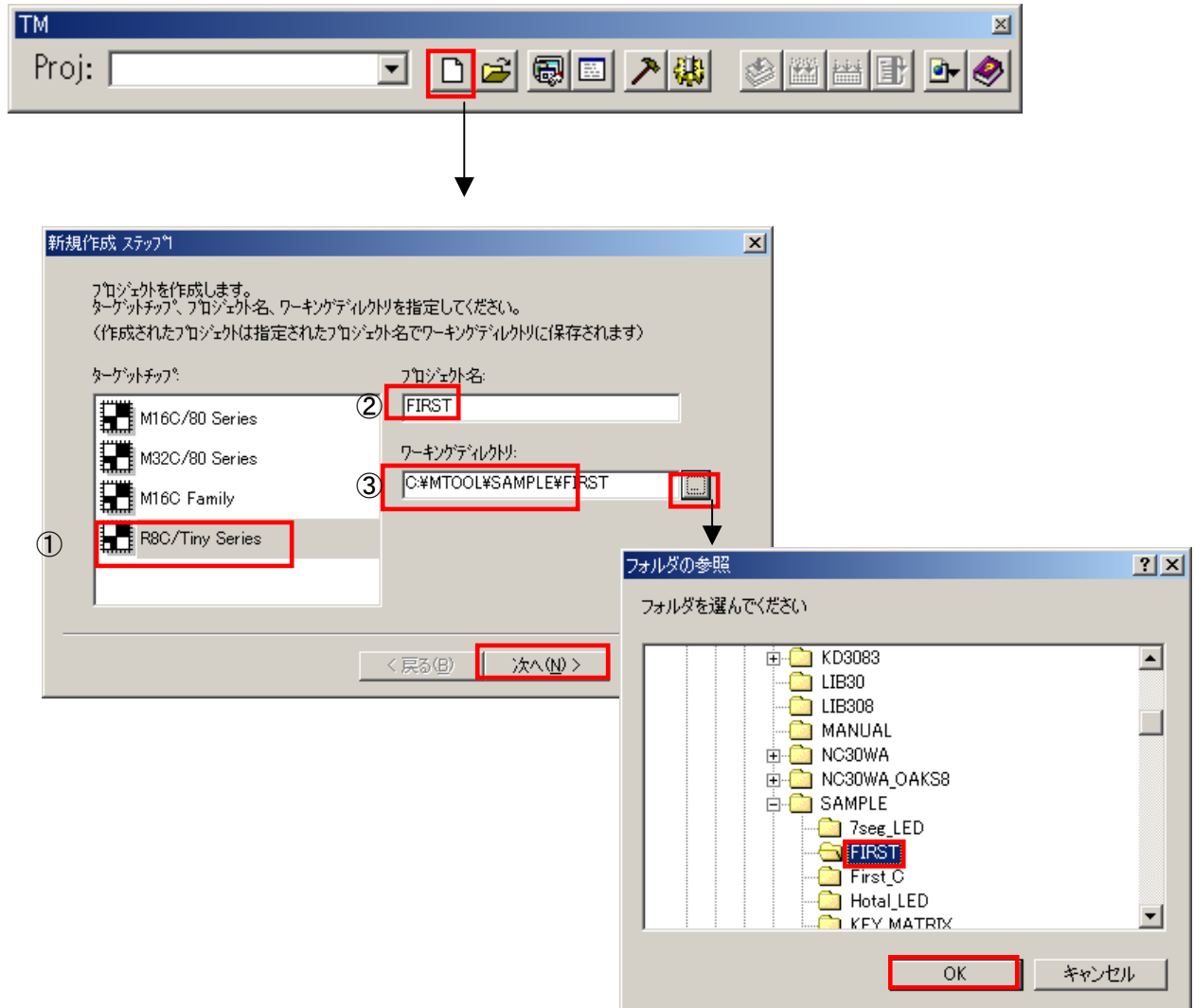


図3.2 プロジェクト新規作成ステップ1

「ステップ2」では「C言語プロジェクト」を選択してください。

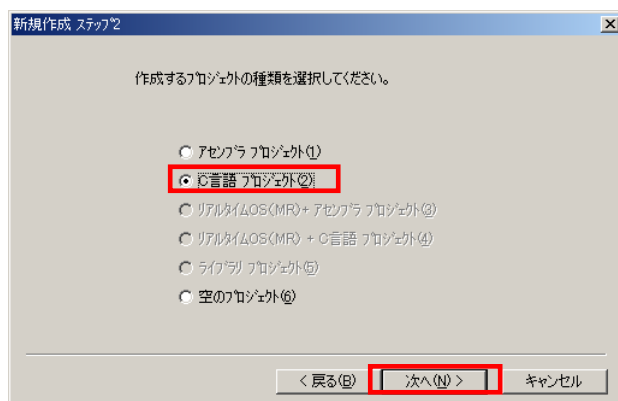


図3.3 プロジェクト新規作成ステップ2

「ステップ-コンパイラ」では以下のように設定してください。

コンパイラパッケージ : NC8 V. 5.30 Release1

スタートアッププログラム : カスタム → FIRSTディレクトリ内のncrt0.a30 (OAKS8専用スタートアッププログラム) を指定

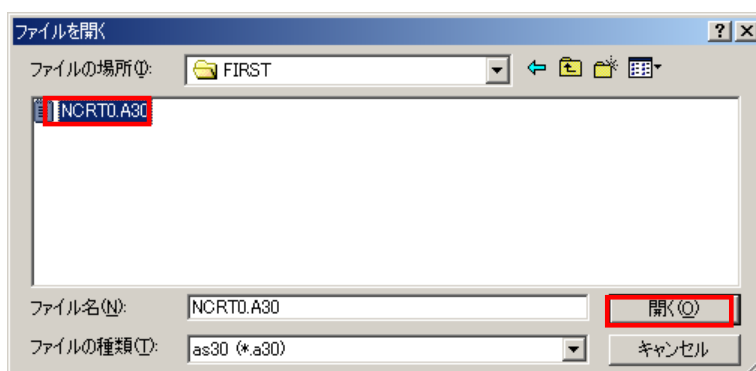
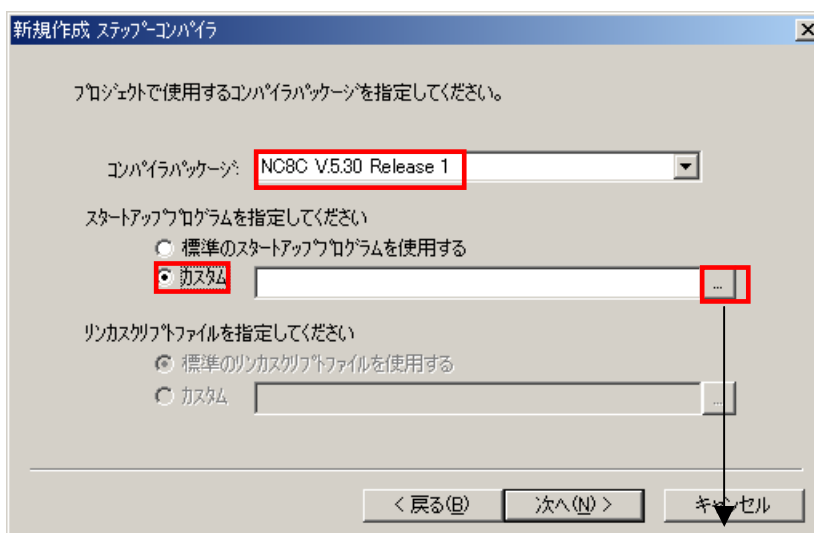


図3.4 プロジェクト新規作成ステップ-コンパイラ

「ステップー完了」の設定項目を確認して、プロジェクト作成の設定を完了します。

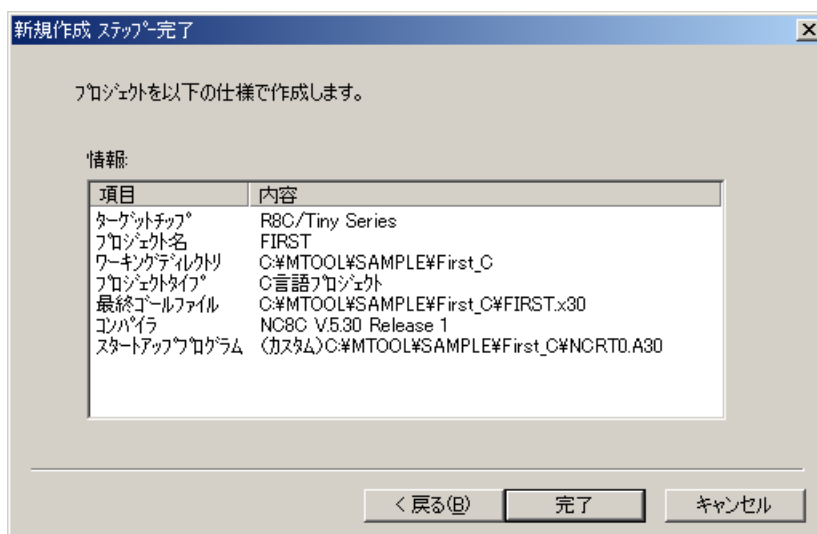


図3.5 プロジェクト新規作成ステップー完了

プロジェクトが完了すると、設定した情報にもとづき、プロジェクトエディタに表示されます。

「プロジェクトエディタ」の左側「生成手順ビュー」に表示されているファイルの左側の「+」ボックスをクリックして、登録されているすべてのファイルを展開してください。

最終ゴールファイルとなる「FIRST.x30」、「FIRST.x30」を生成するコマンドファイル「FIRST.TMK」（ユーザの設定に基づいてTMが生成します）、カスタムスタートアッププログラムとして登録した「NCRT0.A30」が登録されているのがわかります。

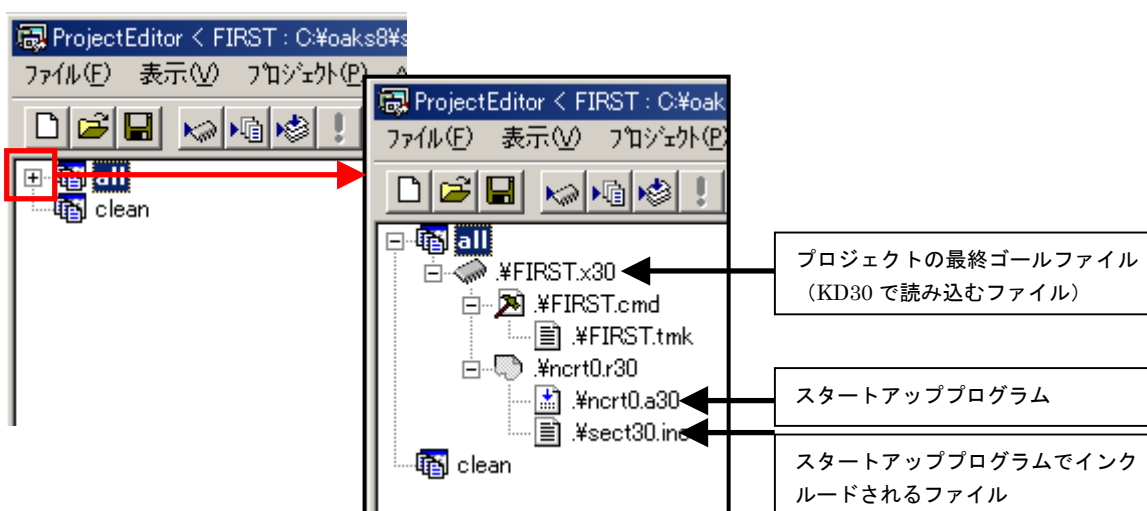


図3.6 プロジェクトに登録されているファイルを展開する

プロジェクトに含まれるファイルの登録、コンパイル/アセンブルオプションの設定などは、プロジェクトエディタで行います。



図3.7 プロジェクトエディタ

3.1.3 リンクするすべてのファイルを登録する

プロジェクトの最終ゴールファイル“FIRST.x30”を選択して、「ファイルの追加」ボタンを押してください。“MAIN.C”ファイルを選んで「開く」ボタンを押してください。プロジェクトエディタで登録されていることを確認してください。

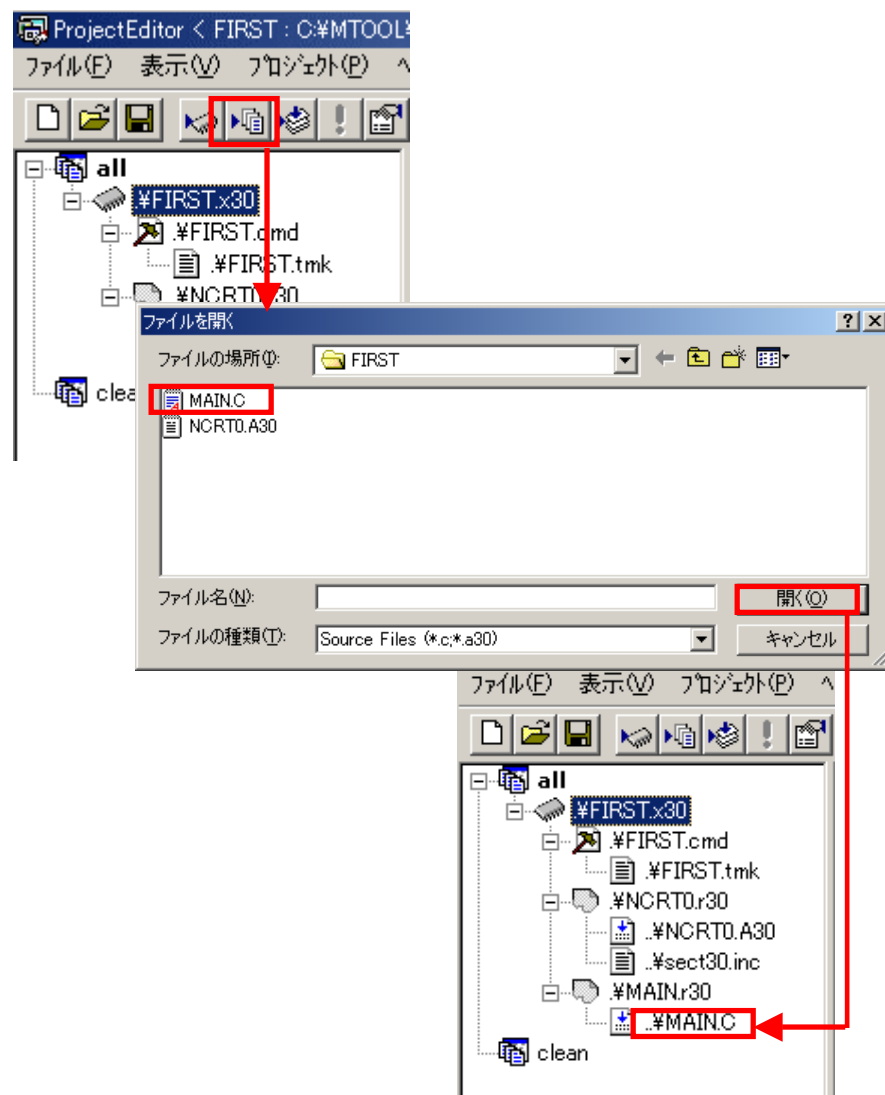


図3.8 ファイルの登録

3.2 プロジェクト作成時に設定されているオプション

NC80コンパイラの場合のみ、基本のオプションに加え、以下のオプションがデフォルトで付けられます。

Cコンパイラオプション → -R8C
アセンブラオプション → -D_R8C_=1
リンカオプション → -L r8c.lib

Cコンパイラ、アセンブラオプションの“-R8C”、“-D_R8C_=1”は、R8C/Tinyシリーズのマイクロコンピュータに対応したコードを生成するためのものですので、外さないでください。

リンカオプションの“-L r8c.lib”は、R8C/Tinyシリーズのライブラリをリンクするオプションです

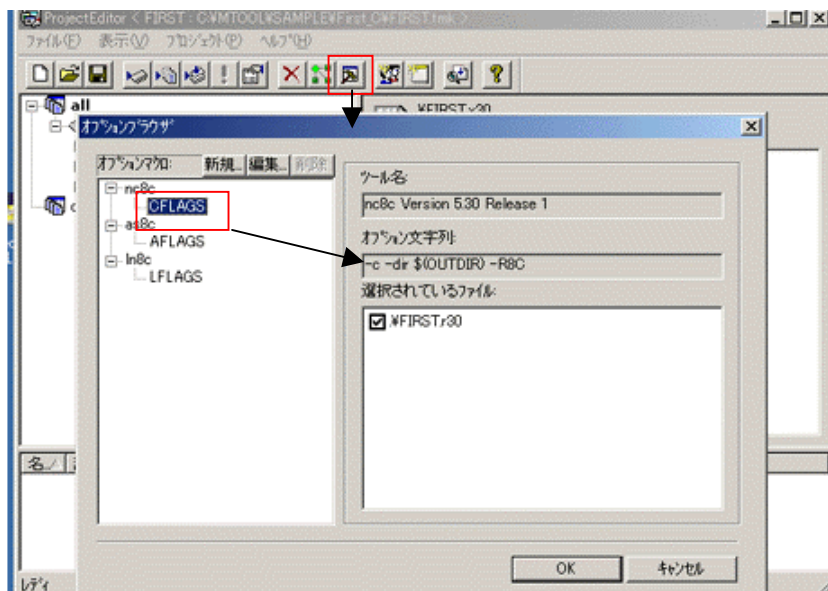
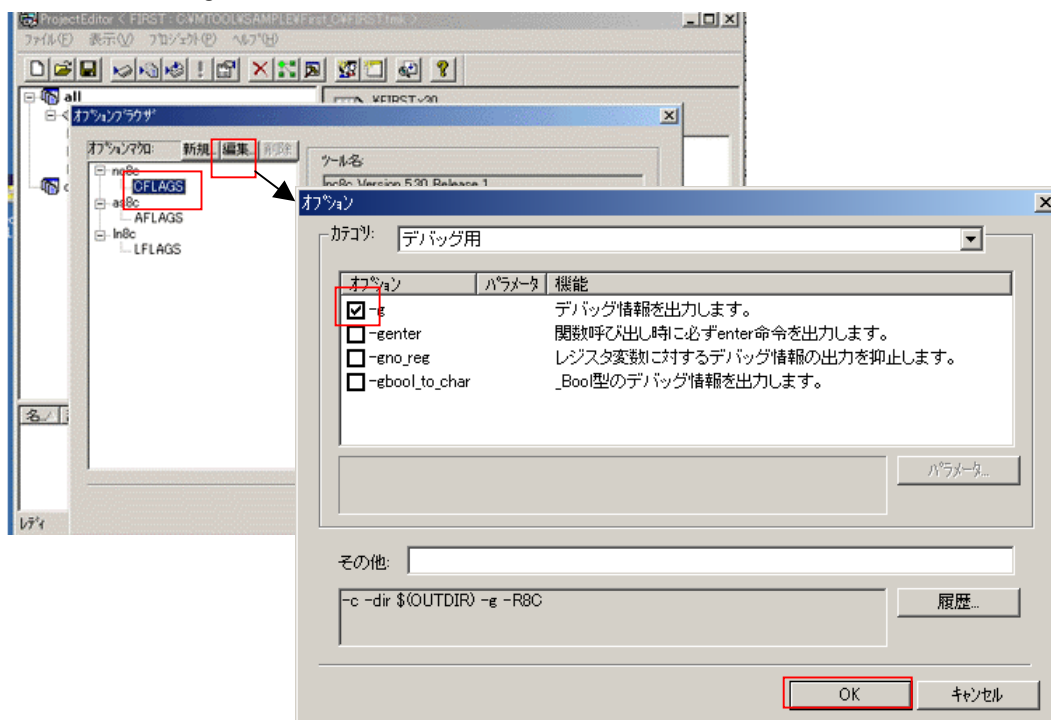


図3.9 オプションブラウザで各オプションを見る

3.3 KD30 でデバッグをする場合に設定が必要なオプション (-g オプション)

KD30でデバッグする場合は、必ず “-g” オプション(デバッグ情報を出力する)を付けてください。コンパイラオプション(CFLAGS)の “デバッグ” カテゴリの “-g” チェックボックスをチェックしてください。



4. Peggy PAD でプログラムを書く

プロジェクトに登録したテンプレートプログラム MAIN.C を完成させます。
ここでは例として、ボード上のLEDをソフトウェアウエイต์で点滅させるプログラムをコーディングします。

4.1 テンプレート「MAIN.C」の内容

プロジェクトエディタに表示されている「MAIN.C」をダブルクリックすると、「Peggy PAD」が開き、「MAIN.C」が表示されます。

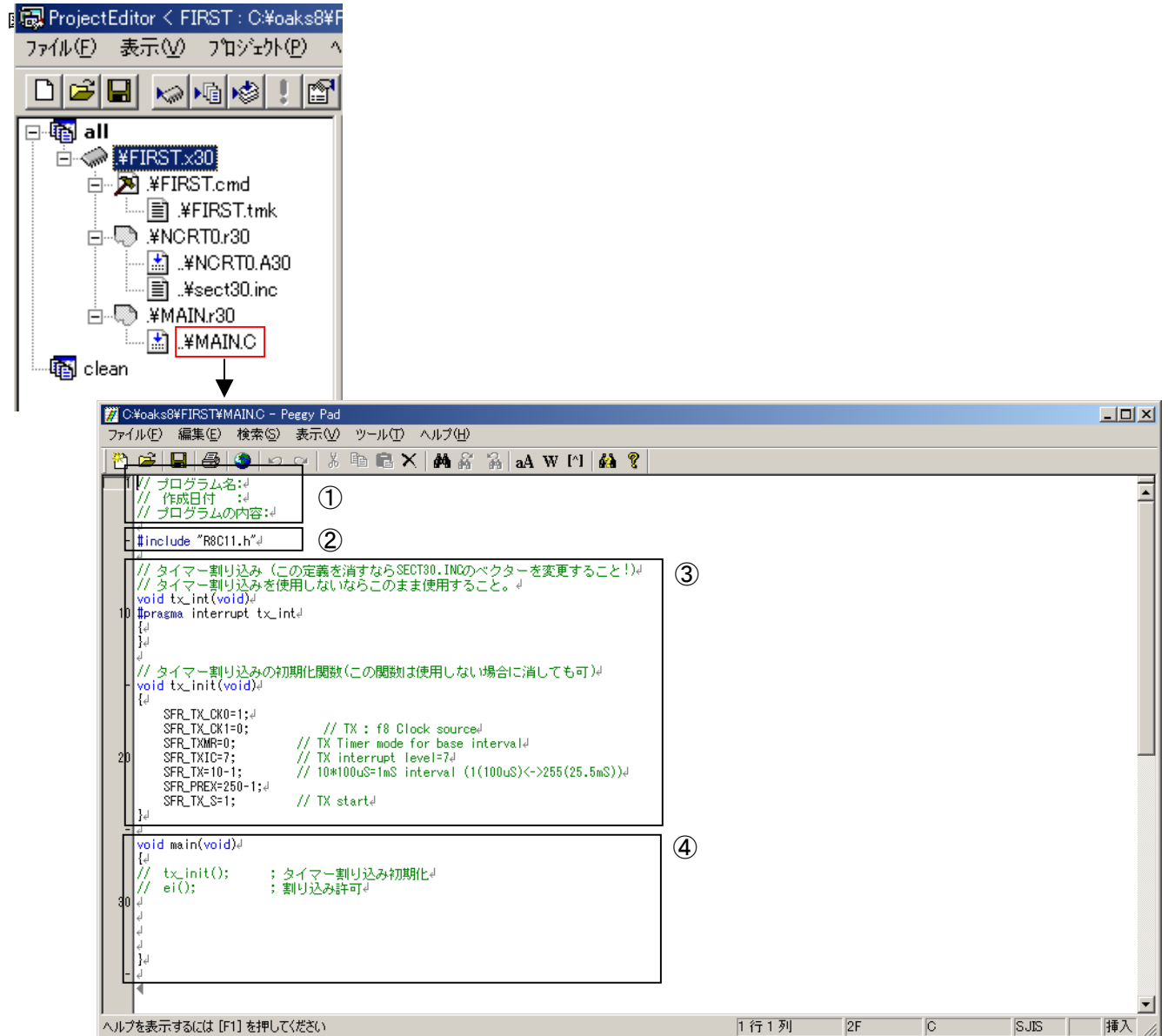


図4.2 テンプレート MAIN.C の内容

① コメント行 (プログラムヘッダ)

プログラムの内容や作成日付、作成者などの情報を先頭に付けておくと便利です。また命令行にもコメントをつけることができます。コメント行はコンパイラでは変換されず、マイクロコンピュータの動作には一切影響を与えないものです。半角英数字、全角英数字、半角カタカナのいずれでも記述できます。NC8コンパイラでは、以下のコメントの書き方をサポートしています。

// で始まる行以下にコメントを書く

/* で始まり、*/で終わる行の中にコメントを書く

ここでは、// から始まる方法で記述しています。

※アセンブリ言語の場合は、; 以降がコメント行となります。

② SFR定義ファイルのインクルード

SFR (マイクロコンピュータの周辺機能専用のレジスタ) の名称を設定しているファイルをインクルードします。プログラム中にSFRを記述する場合は番地を記述しますが、SFRの名称で定義しておくこと、プログラムの可読性が高まります。

③ 割り込みルーチンと割り込みpragma設定

テンプレートには、タイマXの割り込みを使用する場合のpragma設定と、割り込み処理関数 (tx_int)、割り込みの初期設定関数 (tx_init) が用意されています。タイマXの割り込みを使用しない場合は、割り込み初期設定関数 (tx_init) は消してもかまいませんが、割り込み処理関数 (tx_int) の設定は消すとセクション設定ファイル (SECTION30. INC) の割り込みベクタアドレスも変更する必要がありますので、そのまま使用されることをお勧めします。

```
.lword dummy_int          :19 UART1 Tx
.lword dummy_int          :20 UART1 Rx
.lword dummy_int          :21 INT2
.lword tx_int              :22 TimerX
.lword dummy_int          :23 TimerY
.lword dummy_int          :24 TimerZ
.lword dummy_int          :25 INT1
```

MAIN関数のtx_int設定を消去した場合はここをdummy_intに変更する

リスト4.1 SECTION30. INC抜粋

④ メイン関数

メイン関数内の初めの2行は、タイマX割り込みを使用する場合の設定ですが、コメント行になっていますのでこのままでも機械語には変換されません。

タイマXを使用しない場合は、このままか消去してもかまいません。

4.2 Peggy PAD の表示設定をする

半角スペース、全角スペースは見分けがつきにくく、記号で表示させるように設定します。また、コンパイルエラー等は行番号で指定されるので、行番号も表示するように設定します。

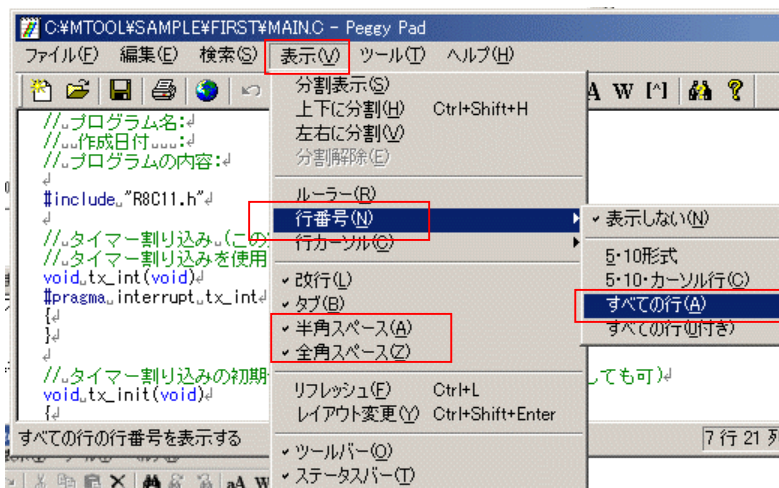


図4.2 Peggy PADの表示設定

4.3 テンプレートにプログラムを書く

練習として、ソフトウェアウエイトのプログラムをメイン関数に書いてみましょう。

リスト4.2 ソフトウェアウエイトのプログラム

<pre>void main(void) { // tx_init(); ; タイマー割り込み初期化 // ei(); ; 割り込み許可</pre>	→	ここはそのままでも消去してもかまわない
<pre> long t; SFR_P4D_5=1; SFR_P4_5=1; while(1) { for (t=0L;t<20000L;t++); SFR_P4_5=SFR_P4_5C^1; } </pre>	→	ここを付け足す

※命令後、数値はすべて半角英数で入力してください。

※空白（空白）は、スペース、タブのどちらでもかまいませんが、全角のスペースは入れないで下さい。

（記述例－空白の入れ方）

- ← 1スペース（スペースキーを一回押す）
- ◀▶ ← 1タブ（タブキーを一回押す）

◀▶	long	□	t;
◀▶	SFR_P4D_5=1;		
◀▶	SFR_P4_5=1;		
◀▶	while(1)		
◀▶	{		
◀▶	for (t=0L;t<20000L;t++);		
◀▶	SFR_P4_5=SFR_P4_5C^1;		
◀▶	}		

リスト4.3 空白の入れ方

5.2 エラーが出たら

エラーメッセージがエラーのある関数名と行数を指定するので、ファイルを開いて該当する行を見直してください。

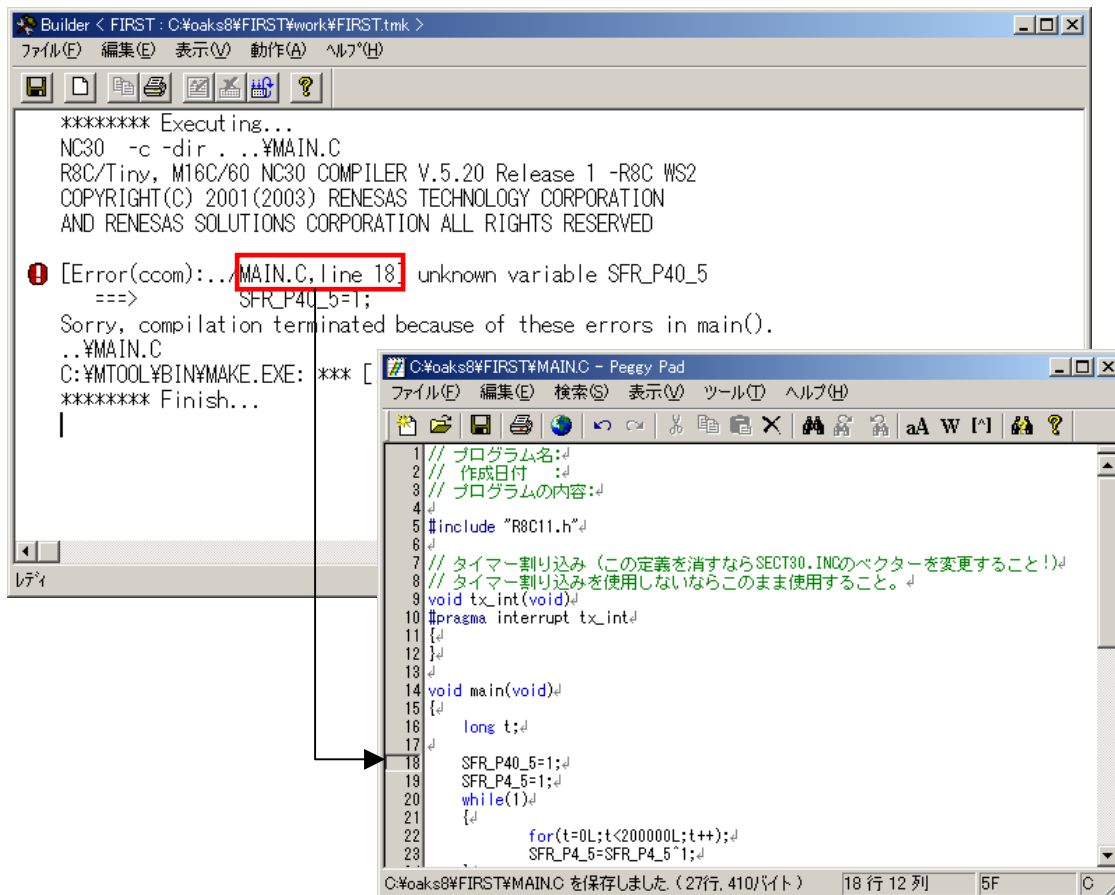


図5.3 「TM」ビルドのエラー

6. KD30 でプログラムを実行する

6.1 KD30 の起動からプログラムの実行まで

OAKS8-EXBOARDとPCを接続し、BOOT端子がショートされているのを確認してから電源を投入してください。

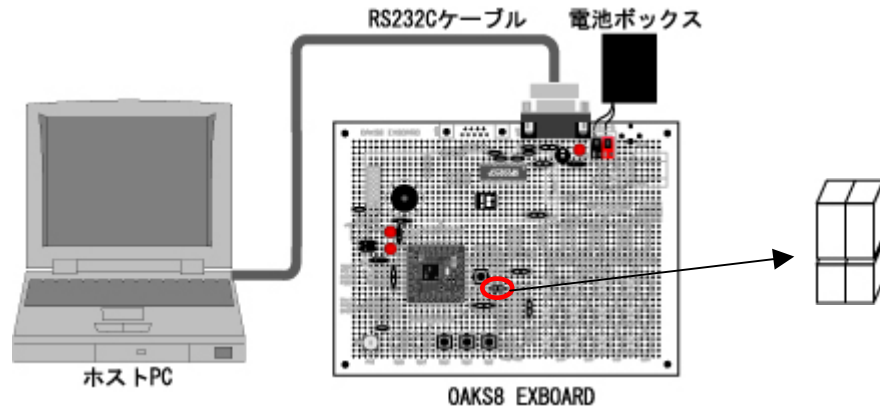


図6.1 OAKS8-EXBOARDとホストPCの接続

TMのプロジェクトバーより「デバッグの起動」をクリックしてKD30を起動してください。



図5.2 TMからのKD30の起動

通信が確立し、プログラムウィンドウが開いたらプログラムをダウンロードします。「File」メニューより「Download」-「Load Module」でプログラムを指定してください。プログラムの実行ファイルはFIRST/WORK/FIRST.x30 です。

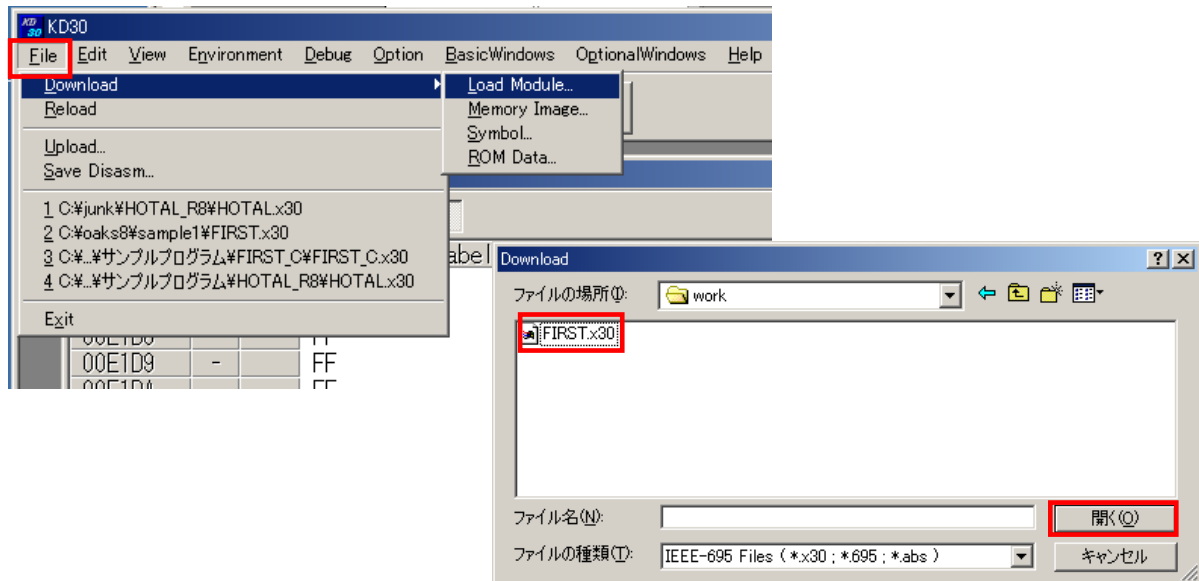


図6.3 KD30-プログラムのダウンロード

プログラムがプログラムウインドウに表示されます。

プログラムの実行

プログラムの停止

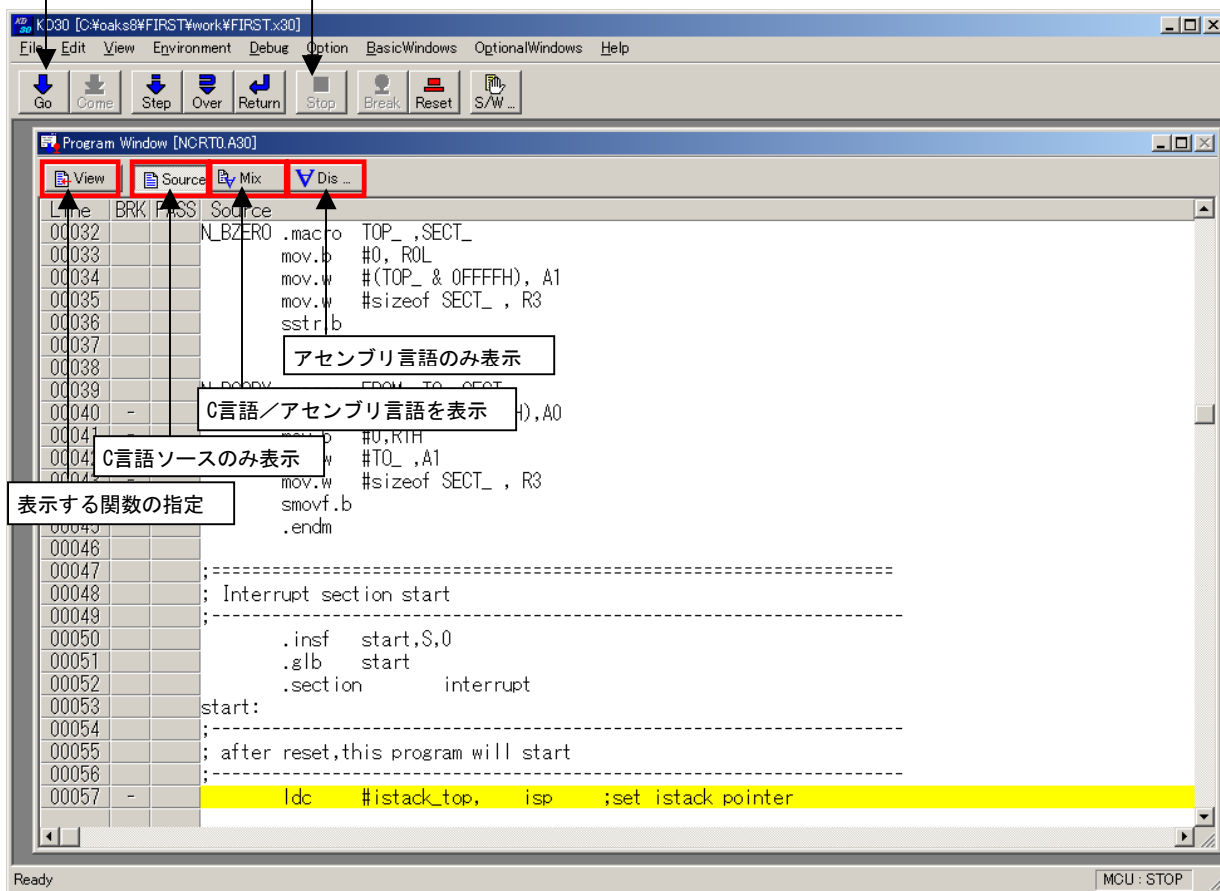
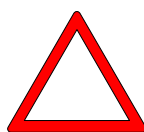


図6.4 KD30-プログラムダウンロード後

「Go」ボタンを押して、プログラムを実行させてください。「Stop」で停止します。

CPUボード上の緑色LEDが点滅します。
プログラムが正常に実行されていることを確認してください。



注意！

フラッシュメモリにプログラムを書き込んでいる最中に、キットの電源を切ったり、通信ケーブルを外したりしないでください。フラッシュメモリに不定のデータが書き込まれ、IDコードがわからなくなる恐れがあります。

7. 実機でプログラムを実行する

KD30でユーザプログラムダウンロード後に、マイクロコンピュータをシングルチップモードで再起動し、ユーザプログラムを実行します。

KD30を終了後、ボードの電源をOFFにしてBOOT端子をオープンに設定してください。再度電源を投入すると、KD30でダウンロードしたプログラムが実行されます。

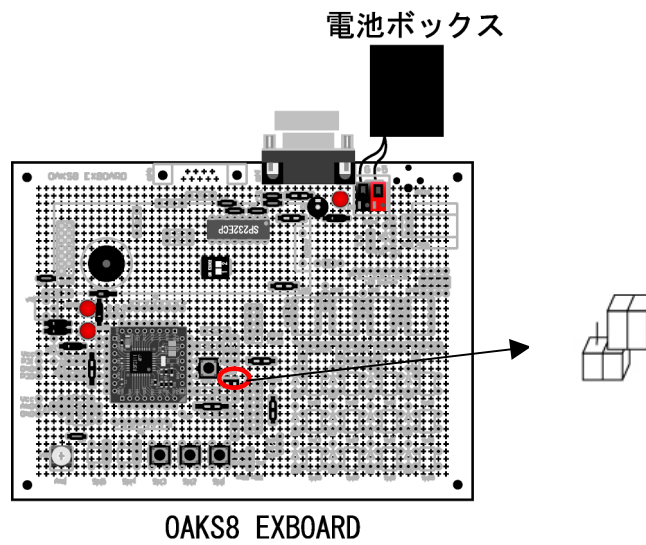


図7.1 OAKS8-EXBOARDと電源の接続

8. Flash Starter でプログラムを書き込む

8.1 TM での設定を追加する

最終的に「Flash Starter」でフラッシュメモリに書き込む際のみ、アブソリュートモジュールファイル（.x30）をモトローラS形式の機械語ファイルに変換します。リロケータブルファイル（.x30）をモトローラS形式機械語ファイル（.mot）に変換するには、コンパイラのロードモジュールコンバータ（LMC）という変換プログラムを使用します。最終ゴールファイルの“FIRST.x30”から“FIRST.mot”に変換するための手順を追加します。

「プロジェクト」メニューの「情報」をポイントして「プロジェクトプロパティ」ウインドウを開きます。「ツール」ウインドウの「パッケージ情報」の「lmc30」のチェックボックスをチェックしてください。

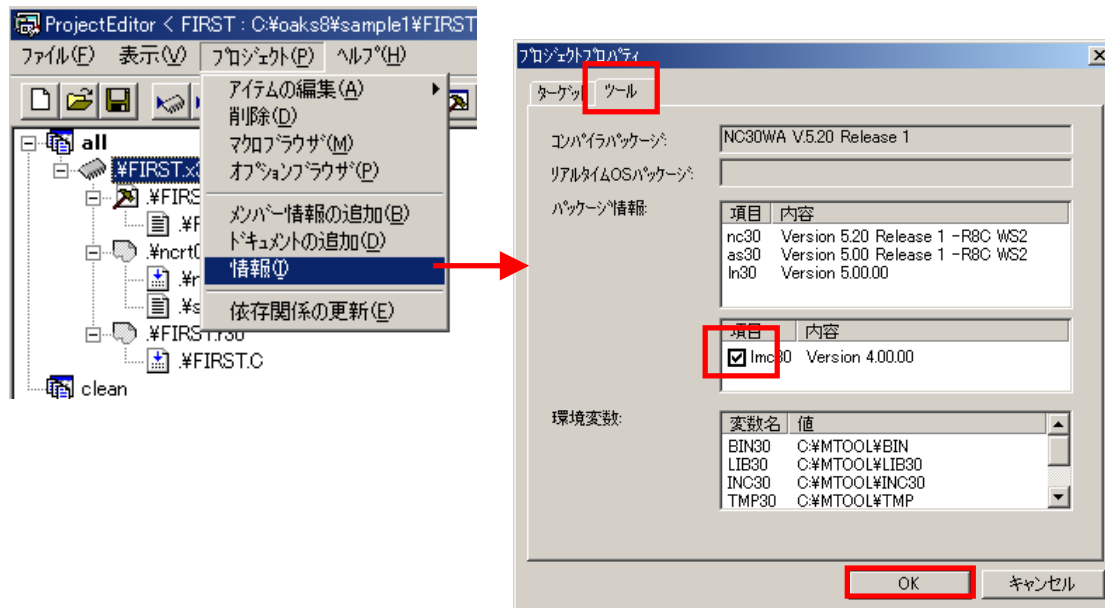


図8.1 LMC30の追加

変換手順の追加を、「プロジェクトエディタ」の右側（アイテム情報ビュー）で確認できます。情報の更新のため、一度他のファイルをクリックしてから再度“FIRST.x30”をクリックしてください。

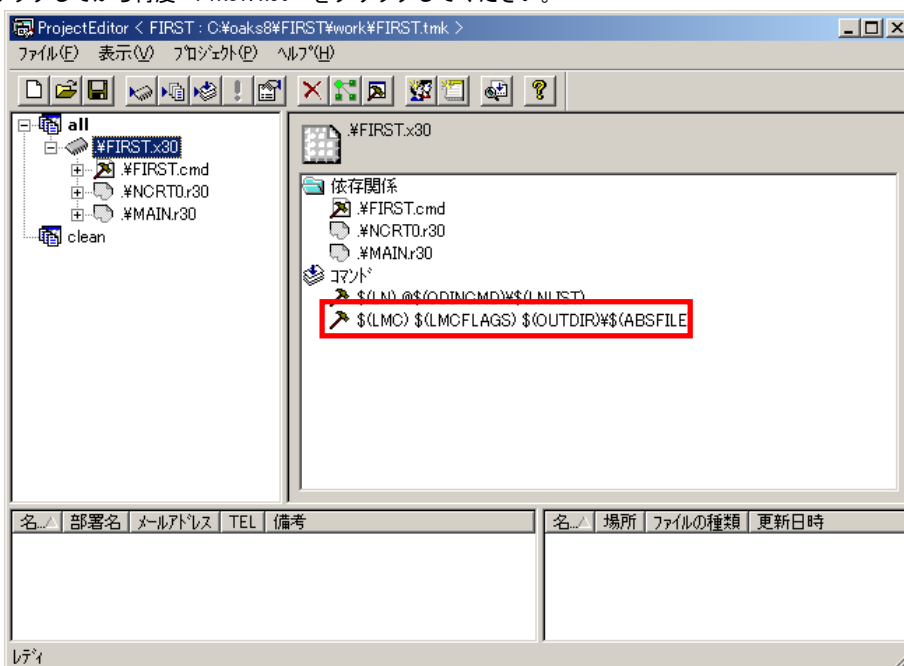


図8.2 LMC30の確認

★IDコードについては、次章の「注意事項 IDコードについて」を参照してください。

LMC30のオプション 「-ID」を指定します。

「-ID」を指定すると、設定した数値をIDコードとしてフラッシュメモリに書き込み、IDファイルを自動的に生成します。

「プロジェクトエディタ」の「オプションブラウザ」ボタンをクリックし、「オプションブラウザ」を表示させます。

「オプションブラウザ」の「LMCFLAGS」をクリックして選択した状態で、「編集」ボタンをクリックします。

「オプションウィンドウ」の「プロテクト制御」カテゴリの中の「-ID」チェックボックスをチェックするとパラメータ入力のウィンドウが開きます。ここではすべてFFhを設定することとします。

パラメータの数値の前には必ず“#”をいれてください。各ウィンドウの「OK」ボタンを押して戻ってください。

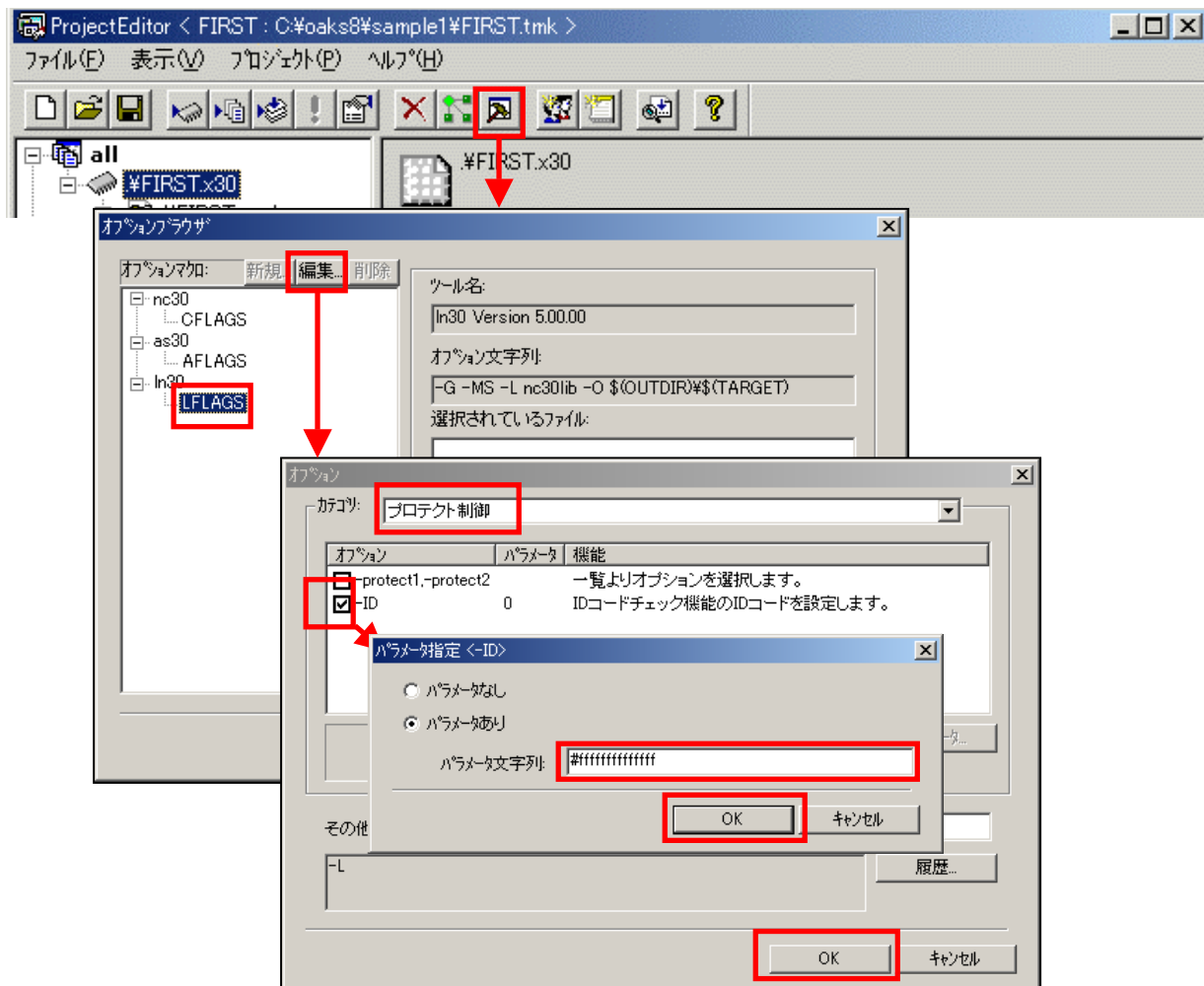


図8.3 IDオプションの設定

「プロジェクトエディタ」の「プロジェクトの上書き保存」をクリックして、設定を保存してください。



図8.4 プロジェクトの保存

ビルドボタンをクリックしてビルドしてください。

8.2 Flash Starter でプログラムを書き込む

① Flash Starterを起動する

KD30を終了した場合は、一度ボードをリセットしてください。

ボードのBOOTピンにジャンパプラグが差し込まれている（ショートされている）のを確認してください。

TMのプロジェクトバーよりFlash StarterのアイコンをクリックしてFlashSta.exeを起動してください。



Flash Starter初期画面が表示されます。この時点では、まだ通信されていません。Port番号を確認して「OK」をクリックしてください。

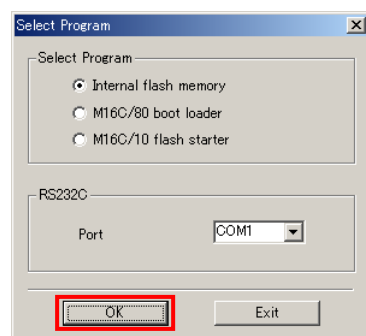


図8.5 Flash Starter初期画面

②書き込むプログラム（機械語ファイル）を指定する。

「OK」を押すと通信が行われ、ファイル指定、ID Checkの設定ウィンドウが表示されます。「MCU Type」はR8Cをチェックしてください。

「Refer」ボタンを押してFIRST.motファイルを選んでください。

ファイルが指定されると同時に、FIRST.idファイルよりIDコードが読み込まれ、IDが設定されます。

「OK」を押してください

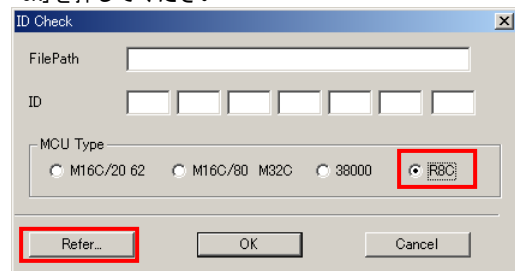


図8.6 Flash Starter ID入力画面

※通信が確立しない場合は、一度ボードの電源を切ってしばらくおいてから再度電源投入、起動してください。

★IDコードの照合が合わない場合は以下の数値を入れなおしてみてください。

全て00h

③プログラムを書き込む。

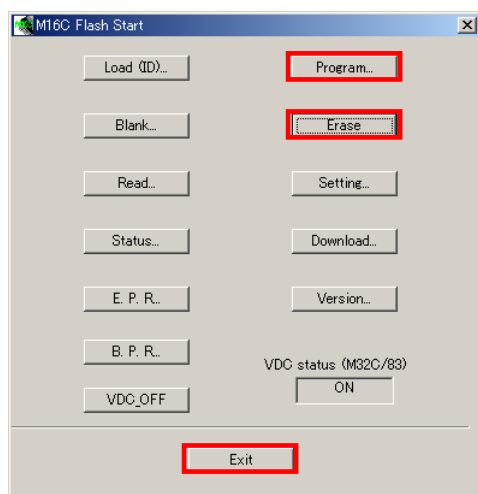


図8.8 Flash Starter メニュー

まず、フラッシュメモリの内容を消去します。「Erase」ボタンを押してください。確認のメッセージが出ますので、「OK」を押してください。

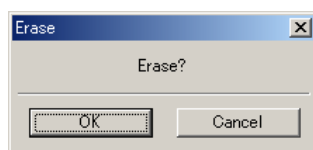
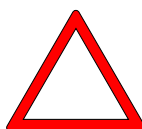


図8.9 Flash Starter 確認メッセージ

「Program」ボタンを押してください。
書き込む領域は自動的に設定されています。「OK」を押してください。

書き込みが終了すると確認のメッセージが出ますので、「OK」を押してください。
「Exit」を押してFlash Starterを終了してください。



注意！

フラッシュメモリにプログラムを書き込んでいる最中に、キットの電源を切ったり、通信ケーブルを外したりしないでください。フラッシュメモリに不定のデータが書き込まれ、IDコードがわからなくなる恐れがあります。

④ 実機で動作させる。
ボードの電源をOFFにし、BOOTのショートプラグを外してオープンにします。
再度電源を投入して、プログラムが動作するか確認してください。

9. KD30 と Flash Starter の動作について

9.1 KD30 について

8.1.1 KD30 について

KD30は、ホストPC側のプログラム (KD30.exe) と、マイコン内蔵のフラッシュメモリに書き込まれるR8C/Tinyシリーズ用モニタプログラム (R5F21114UART.s : モトローラS形式機械語ファイル) の2つのプログラムで構成されます。ホストPC側のプログラムがモニタプログラムへコマンドを送信し、モニタプログラムがコマンドに応じてマイクロコンピュータを操作したり、マイクロコンピュータの状態をホストPC側のプログラムへ送信します。このような方法でデバッグを実現するものを、リモートデバッグと呼びます。複雑なH/Wを必要とせずに評価、学習には十分な機能を備えています。

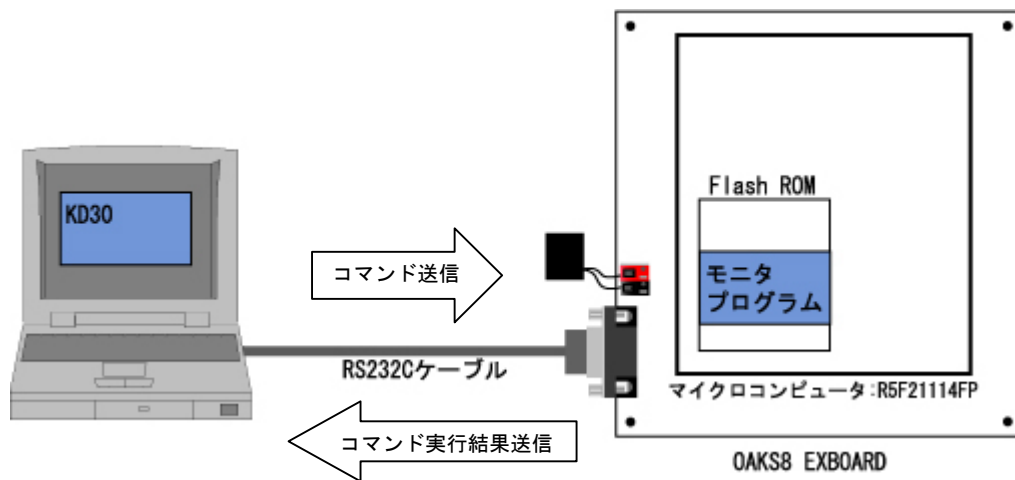


図9.1 KD30コマンド実行 (イメージ図)

※リモートデバッグ方式では、モニタプログラムとユーザプログラムがマイクロコンピュータの資源を共有することになります。したがってKD30でデバッグ時にはユーザプログラムに制限事項が発生します。詳細は「KD30 UART使用時の注意事項.pdf」を参照ください。

KD30はC言語で書かれたプログラムを、ソースコードを表示させながらデバッグすることができます (C言語ソースレベルデバッグ)。C言語ソースレベルでのデバッグを行うためには、コンパイラNC8で、コンパイラオプション“-g” (デバッグ情報をアセンブリ言語ソースファイルに出力する) を付けてコンパイルする必要があります。

又、KD30はルネサス テクノロジ社製のM16Cファミリ対応エミュレータと操作面で互換性があり、製品移行もスムーズに行えます。

9.1.2 KD30 起動時のマイクロコンピュータの動作モードについて

KD30 は必ずマイクロコンピュータが「標準シリアル入出力モード2」の状態起動してください。「標準シリアル入出力モード」とは、マイクロコンピュータ内蔵のフラッシュROMを書き換えるためのブートプログラム（通常のROM領域とは別の領域に工場出荷時にマイクロコンピュータに予め書き込まれているプログラム）を動作させ、外部シリアルライターと通信することによってフラッシュROMを書き換えるためのモードです。

OAKS8に搭載のR8C/Tinyシリーズマイクロコンピュータの場合、CNVss端子、MODE端子が共に“L”の状態ではリセットをかけると、「標準シリアル入出力モード2」となりブートプログラムを実行します。（通常のシングルチップモードではリセットベクタに設定されているアドレスよりプログラムを実行します。）

OAKS8-EXBOARD上でCNVss端子については“L”に設定済みです。残りのMODE端子はOAKS8-EXBOARDのBOOT端子に接続されており、BOOT端子をショート（短絡）すると“L”に設定されます。

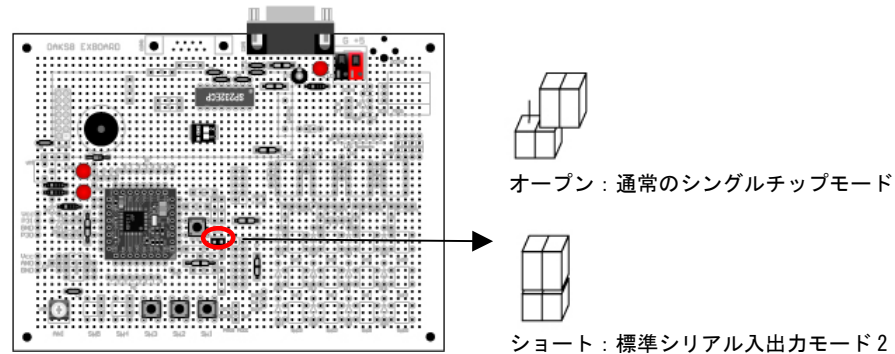


図9.3 OAKS8-FullKitのBOOT端子

9.1.3 KD30の動作について

KD30のホストPC側のプログラムがマイクロコンピュータのブートプログラムと通信を行います。ブートプログラムは通信相手がKD30の場合、現在フラッシュメモリに設定されているIDコードがALL “00h” か “FFh” である場合の時のみ、ホストPCのKD30.exeと同じディレクトリにインストールされているモニタプログラム（R5F21114UART.s）をフラッシュメモリに書き込みます。（起動する毎に必ず書き込みます）

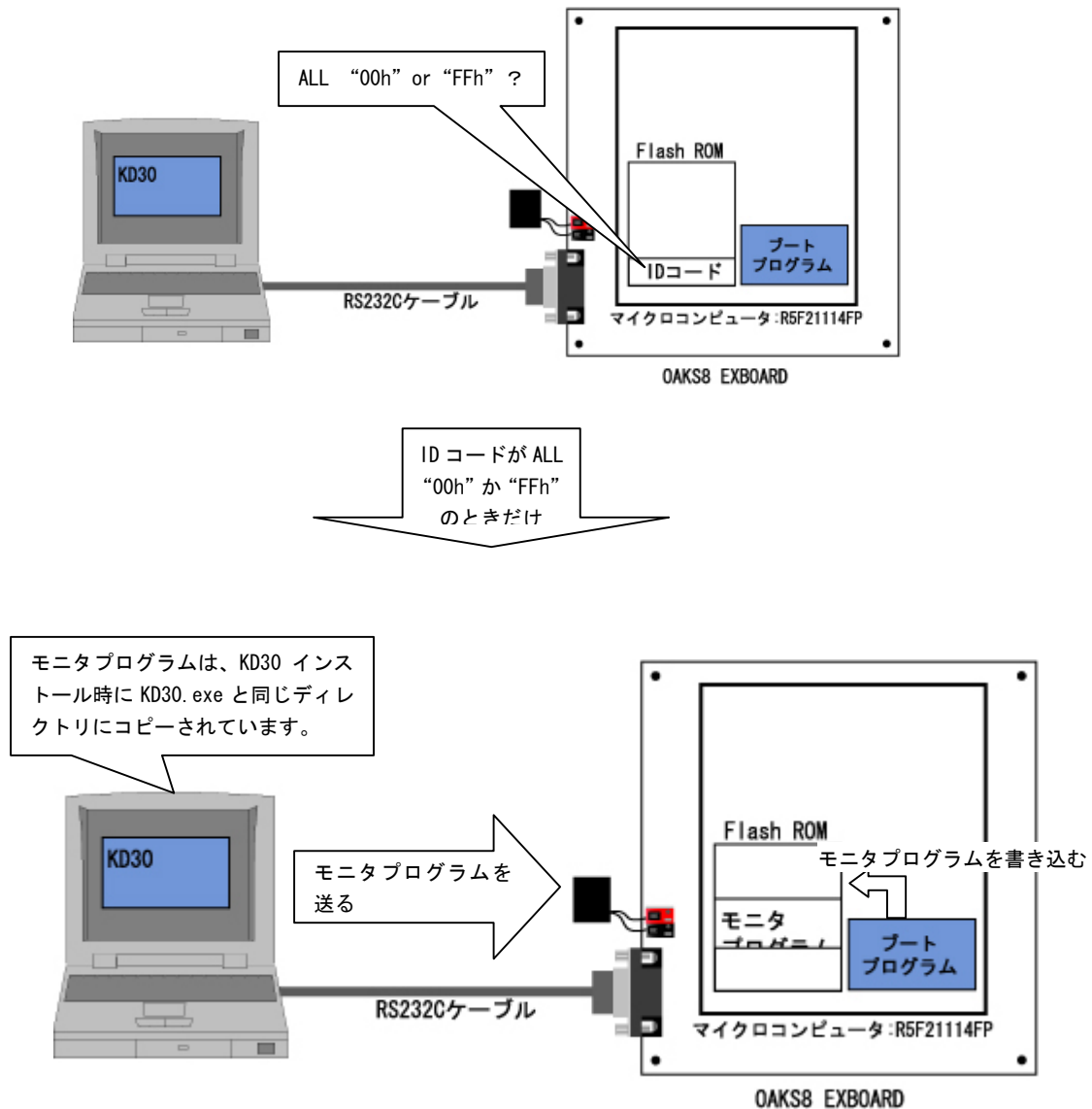


図9.4 KD30起動時—ブートプログラムによるモニタプログラムの書き込み

KD30起動時にフラッシュメモリに設定されているIDコードがALL “00h” あるいは “FFh” でない場合は、ブートプログラムは何もしません。

ホストPC側では、モニタプログラム書き込みのプログレスバーが途中で止まったままの表示になります。この場合は **[Ctrl+Alt+Delete]** キーを押して、KD30を強制終了させてください。

Flash Starterでフラッシュメモリを消去する（IDコードはALL “FFh” となります）などしてから、再度KD30を起動してください。

フラッシュメモリにモニタプログラムが書き込まれると、モニタプログラムが実行され、KD30との通信を始めます。

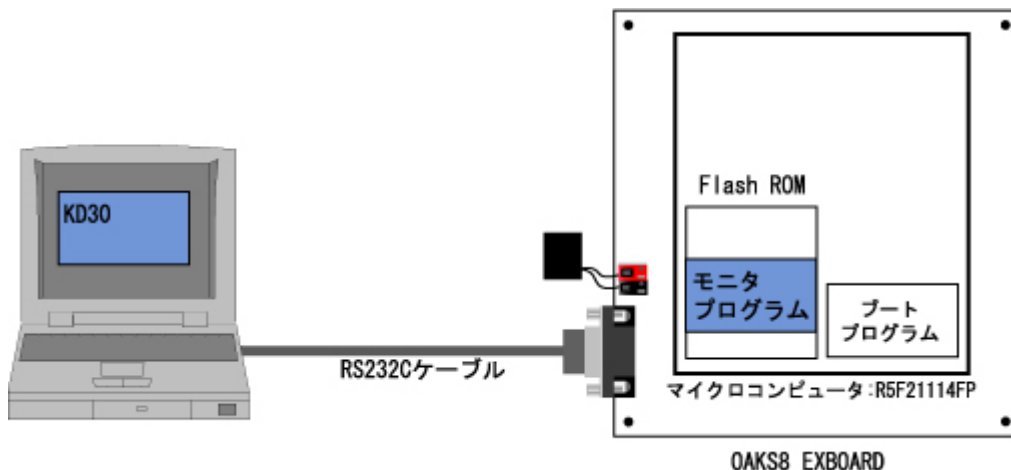


図9.5 KD30起動時—KD30とモニタプログラムの通信

※マイクロコンピュータのブートプログラムを実行させるためには、OAKS8-EXBOARD上のBOOT端子をショートさせてください。
※KD30を一度終了してから再度起動するには、マイクロコンピュータをリセットする必要があります。

9.1.4 KD30 終了後の実機動作について

KD30のモニタプログラムは、マイクロコンピュータのブートプログラムにより書き込まれ、起動されますので、リセットペクタは使用しません。リセットペクタにはユーザプログラムの先頭アドレスを設定できます。KD30でユーザプログラムをダウンロードしたあと、通常のシングルチップモードでマイクロコンピュータを再起動させると、ユーザプログラムを実機で動作させることができます。

9.1.5 他の OAKS シリーズの KD30 が既にインストールされている場合

既にOAKS16シリーズのKD30がホストPCにインストールされている場合は、アンインストールし、OAKS8-FullKitに付属のKD30 Ver4.00 Release 1にバージョンアップしてください。（モニタプログラムが必要なシリーズは、モニタプログラムも更新してください。）

（Windowsの[スタート]メニュー→[設定]→[コントロールパネル]→[アプリケーションの追加と削除]より削除してください）
OAKS8-FullKitに付属のKD30 4.00 Release 1 はOAKS16シリーズのCPUにも対応しています。

9.2 Flash Starter について

Flash Starterは、モトローラS形式機械語ファイル (.mot) に変換されたユーザプログラムをフラッシュメモリに書き込むソフトウェアです。KD30でデバッグしたあと、最終的にシステムにプログラムを組み込む時にFlash Starterを使用して書き込みます。

9.2.1 Flash Starter の動作モードについて

Flash Starterは必ずマイクロコンピュータが「標準シリアル入出力モード2」の状態²⁾で起動してください。「標準シリアル入出力モード」とは、マイクロコンピュータ内蔵のフラッシュROMを書き換えるためのブートプログラム（通常のROM領域とは別の領域に工場出荷時にマイクロコンピュータに予め書き込まれているプログラム）を動作させ、外部シリアルライターと通信することによってフラッシュROMを書き換えるためのモードです。

OAKS8-FullKitに搭載のR8C/Tinyシリーズマイクロコンピュータの場合、CNVss端子、MODE端子が共に“L”の状態²⁾でリセットをかけると、「標準シリアル入出力モード2」となりブートプログラムを実行します。（通常のシングルチップモードではリセットベクタに設定されているアドレスよりプログラムを実行します。）

OAKS8-FullKit上でCNVss端子については“L”に設定済みです。残りのBOOT端子をショート（短絡）させてリセットをかけることで「標準シリアル入出力モード2」となります。

。

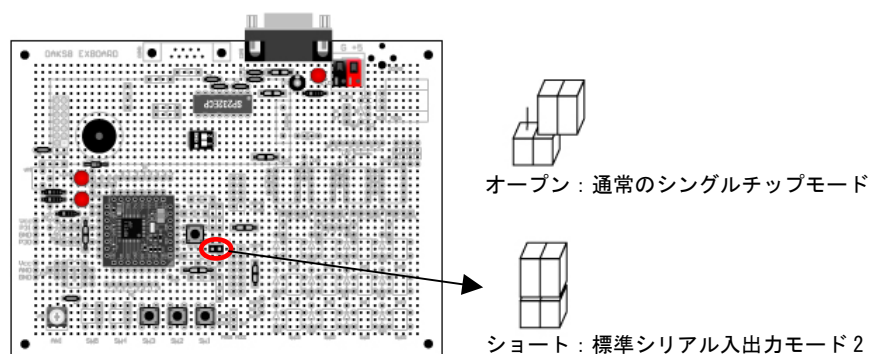


図9.6 OAKS8-EXBOARDのBOOT端子

9.2.2 Flash Starter の動作について

ホストPCのFlash Starterプログラムがマイクロコンピュータのブートプログラムと通信を行います。ブートプログラムは通信相手がFlash Starterの場合、IDコードの照合を要求します。IDコードが合うと、Flash Starterからのコマンドに従ってフラッシュメモリの消去、ユーザプログラムの書き込みなどを行い、結果をFlash Starterへ送信します。

IDコードが合わない場合、フラッシュメモリの消去、書き込みはできません。

※IDコードについては「9章 IDコード」を参照してください。

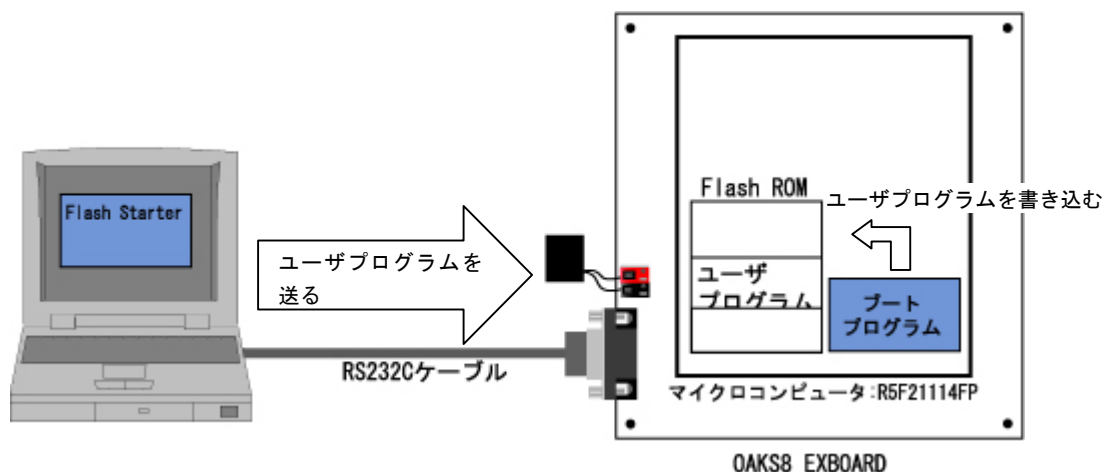


図9.7 Flash Starterのプログラム書き込み（イメージ図）

10. IDコード

10.1 IDコードについて

10.1.1 IDコードとは

IDコードとは、フラッシュメモリへの不正なアクセスを防ぐためのプロテクトのための7バイトのデータです。IDコードは以下のアドレスにユーザが設定します。(半導体工場出荷時はAll“00”です。なにも設定しないでプログラムを書き込んだ場合はAll“FFh”となります)

IDコード格納アドレス : 0FFDFh, 0FFE3h, 0FFEBh, 0FFEFh, 0FFF3h, 0FFF7h, 0FFFBh



図10.1 IDコードの領域

フラッシュメモリ書き込みソフトウェア「Flash Starter」では、フラッシュメモリにプログラムを書き込んだり消去する際に、現在フラッシュメモリに書き込まれているIDコードの照合を要求します。IDコードが合わない場合、「Flash Starter」によるプログラム書き込み、消去はできません。

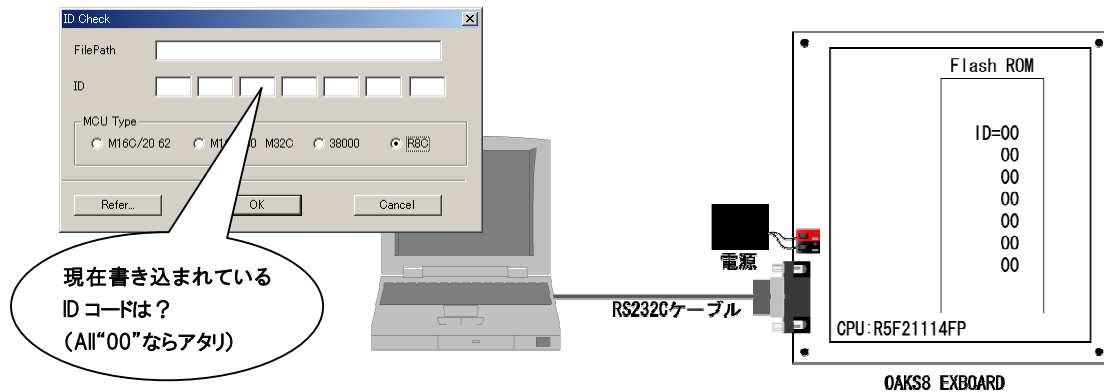


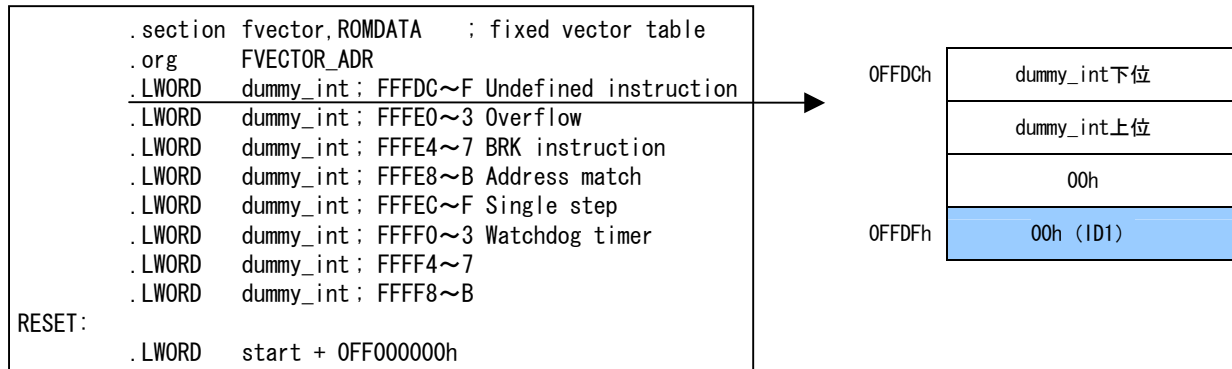
図10.2 IDコード (All“0”の場合) の照合 (イメージ図)

10.1.2 IDコードの設定例

IDコードの設定例を示します。

- ① プログラム内で設定する（リロケータブルファイル .x30 と機械語ファイル .mot に反映）
アセンブラ指示命令で、ベクタアドレスの設定と同時に“00h”を設定します。

sect30.incより抜粋（サンプルプログラムのスタートアッププログラムでインクルードされます）



★ 固定データ設定指示命令 LWORD について
4バイト確保して2バイトデータを設定すると、上位2バイトには“00h”が格納される。

dummy_intは2バイトアドレスなので、上位の2バイトにはそれぞれ“00h”が格納されます。
その結果、IDコード格納アドレスに“00h”が格納されることになります。

- ② LMC（ロードモジュールコンバータ）のIDオプションで設定する（機械語ファイル .mot のみに反映）

リロケータブルファイル（.x30）からモトローラS形式の機械語ファイルへの変換プログラム LMCのIDオプションでIDコードを指定することができます。

IDコードを指定してファイル変換を行うと、機械語ファイルのIDコード格納アドレスに指定したIDコードが書き込まれ、フラッシュメモリに書き込まれることになります。

プログラム内でIDコードを設定していた場合でも、IDオプションで指定したIDコードが上書きされます。

又、IDオプションを指定するとIDファイルも同時に生成します。IDファイルとはモトローラS形式の機械語ファイルと同じファイル名で、拡張として.idが付きます。IDファイルにはIDオプションで指定されたIDコードが記述されます。

「Flash Starter」で機械語ファイルを指定した際、同一ディレクトリ内にIDファイルがある場合は、「FlashStart」のIDコード入力枠に自動的にIDファイルの値が入力されます。

（ただし、IDファイルは現在フラッシュメモリに書き込まれているIDコードと、今回書き込もうとするIDコードが同一の場合に有効です。）

例：IDオプションにALL“0”を設定してファイル変換を行う。 → lmc30 ファイル名 -id #0

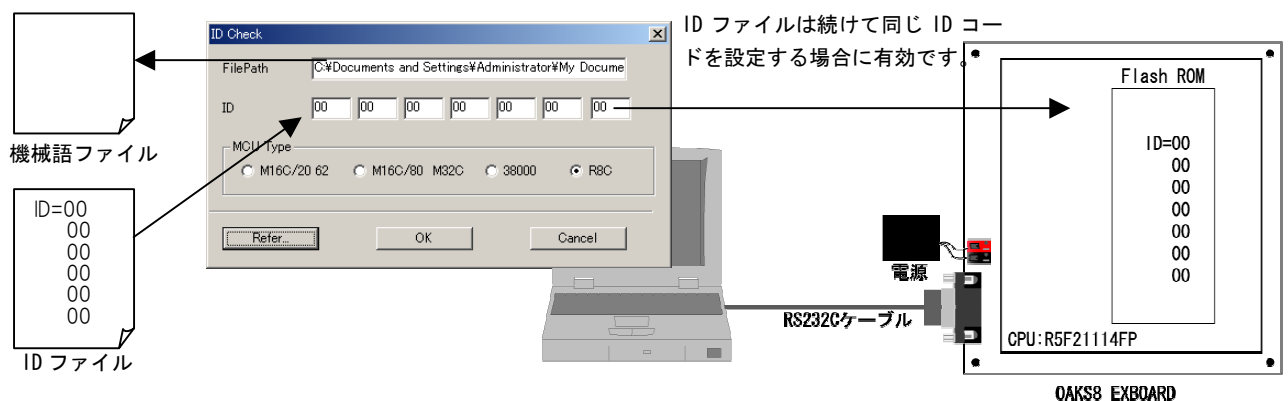


図10.3 IDファイルによるIDコードの照合

★現在フラッシュメモリに書き込まれているIDコードと異なるIDコードを書き込む際には、手入力でのID入力をおこなってください。

10.2 KD30 モニタプログラムが設定する ID コード（重要）

KD30モニタプログラムではアドレス一致割り込み、シングルステップ割り込みと予約ベクタのベクタテーブルの3テーブルを使用しており、また全てのIDコードはFFhに設定しています。

KD30起動後は必ず全てのIDコードがFFhとなっていますので、Flash StarterでIDコードの照合を行う際にはご注意ください。混乱を避けるために、コンパイル時にIDオプションを設定する場合は、全てFFhを設定することをお勧めします。

は、モニタプログラムが設定するID番号です。 は、モニタプログラムが使用するベクタ領域です。

0FFDCh	未定義命令ベクタ	0FFE8h	アドレス一致ベクタ	0FFF4h	予約
	未定義命令ベクタ		アドレス一致ベクタ		予約
0FFDFh	ID1=FFh	0FFEBh	ID3=FFh	0FFF7h	ID6=FFh
0FFE0h	オーバーフローベクタ	0FFECh	シングルステップベクタ	0FFF8h	予約
	オーバーフローベクタ		シングルステップベクタ		予約
0FFE3h	ID2=FFh	0FFEFh	ID4=FFh	0FFFBh	ID7=FFh
0FFE4h	BRK命令ベクタ	0FFF0h	監視タイマベクタ	0FFFCh	リセットベクタ
	BRK命令ベクタ		監視タイマベクタ		リセットベクタ
		0FFF3h	ID5=FFh	0FFFh	

図10.4 KD30が使用するベクタと、設定するIDコード

10.3 現在書き込まれている ID コードがわからなくなったら（重要）

以下の方法で解決してください。

10.3.1 機械語ファイルを参照する

FlashStartで書き込んだ場合は現在書き込まれているプログラムの機械語ファイル（.mot）より、IDコードを参照してください。

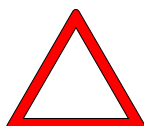
例：サンプルプログラム FIRST.mot （LMCオプション -ID #12345678123456 でビルド）

S0030000FC	
S22400	E000 7CF2047E9F55077E9F4507D9106A3AD90BFCD90BFED10BFE7DCC137DCE08778BCD
S22400	E020 FC204E6809C91BFC77EBFEFEE9B47EBF45077E2000FA7204D910890472407EB038
S20E00	E040 007E2F4507FAFEC47DF2AD
S22400	E04A EB400008C7010A00C7480600C7280700B70C00C7080600B70A00EB600004EB20EE
S22400	E06A 0000EB1000FFA200E0D803AA000475C300007CE8A200E0D803AA000475C30000AD
S22400	E08A 7CE8A200E0D803AA000475C300007CE8A200E0D803AA000475C300007CE8FD00C2
S20900	E0AA E000FEFFFB94
S22400	FF00 AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE000006C
S22400	FF20 AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE000004C
S22400	FF40 AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE000002C
S22400	FF60 AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE000000C
S22400	FF80 AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000EC
S22400	FFA0 AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000CC
S22000	FFC0 AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE00000AEE000003E
S22400	FFDC AEE00012AEE00034AEE00000AEE00056AEE00078AEE00012AEE00034AEE0005660
S20800	FFFC 4AE000FFD3
S804000000FB	

緑色はモトローラフォーマットのレコードタイプやチェックサムデータ

青色の囲みは機械語が格納されるアドレス

赤字はIDコード



注意！

ユーザプログラム、あるいはIDオプションで設定したIDコードの扱いにおいては、ご注意ください。IDコードがわからなくなってしまうと、Flash Starterでの消去、書き込みができなくなり、最悪の場合KD30、Flash Starterが使用できなくなります。

10.4 付録のサンプルプログラムでの ID コード設定

OAKS8キットに付属のサンプルプログラム（オークス電子株式会社サイトからのダウンロードを含む）では、プログラムでは全てのベクタアドレスを設定するとともに、IDコード領域にはを00hに設定しています。またTMプロジェクトで、LMCのIDオプションでIDを全てFFhに設定していますので、結果的に、生成されるxx.motファイルのIDコードはすべてFFhが設定されます。

11. 制限事項

主にKD30の使用において発生するマイクロコンピュータの機能制限について説明します。

11.1 メモリ制限

以下にKD30使用時のメモリマップを示します。ユーザが使用できる領域は以下のとおりです。

RAM : 400h~7FFh (制限なし)

ROM : C800~FFFFh (ただし、ベクタ領域、IDコードの一部に設定します)

C000h~C7FFhまではモニタプログラムの領域となりますので、この領域にプログラムを書き込まないで下さい。KD30は、ユーザプログラムがモニタプログラムと重なった場合、モニタプログラムと重なった領域は書き込みません。また、この場合KD30はメッセージ等を出しませんのでご注意ください。

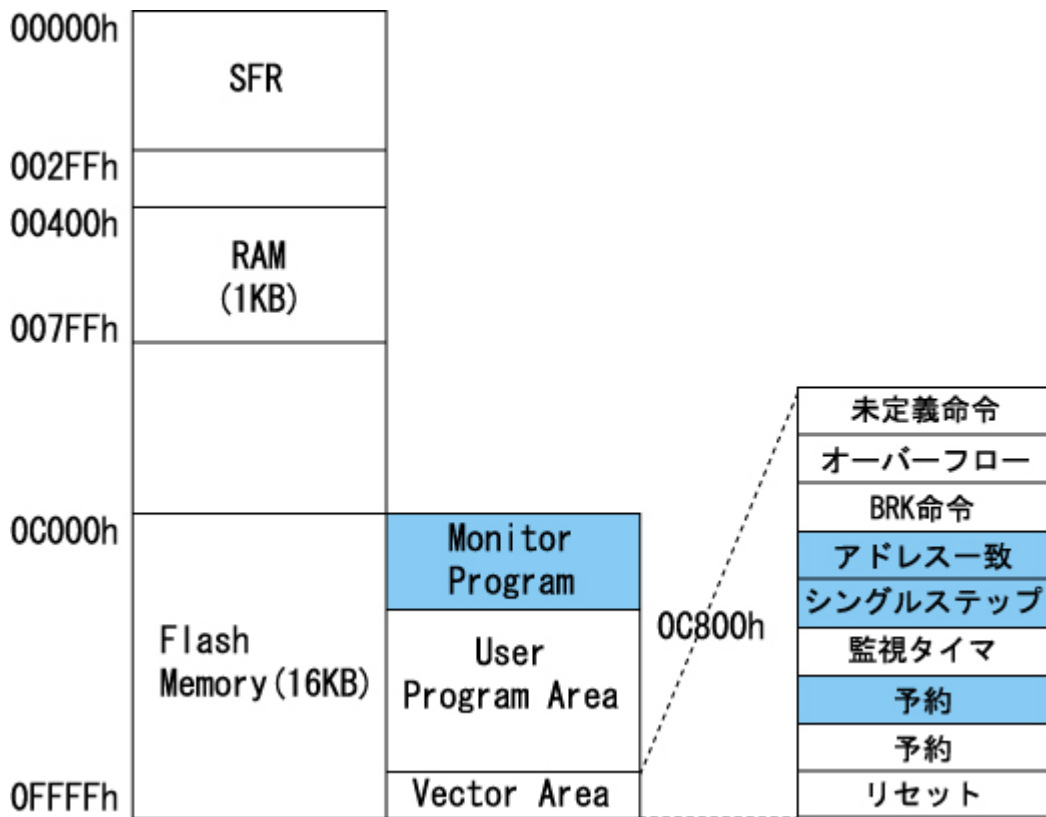


図11.1 KD30使用時のメモリマップ

11.2 レジスタ操作に関する制限

以下のレジスタはKD30モニタプログラムによって設定、あるいは使用されます。変更が禁止されているレジスタを変更した場合、モニタプログラムは正常に動作しませんのでご注意ください。
 モニタプログラムはマイクロコンピュータのUART1を使用して、ホストPCと通信を行いますので、UART1関連のレジスタの値を変更しないようにしてください。

表11.1 KD30が使用/設定するレジスタ

レジスタ名	KD30モニタプログラムが設定する値	制限事項	変更
プロセッサモードレジスタ0	00h	シングルチップモードでのみ動作します	△
プロセッサモードレジスタ1	00h		○
システムクロック制御レジスタ0	08h		○
システムクロック制御レジスタ1	28h	XIN-XOUT駆動能力ビットは“1”にしてください。	△
高速リング制御レジスタ0	03h		○
高速リング制御レジスタ1	7Fh		○
発振停止検出レジスタ	00h		○
プロテクトレジスタ	—		○
フラグレジスタ	—	Dフラグへの書き込みは無視されます。ユーザプログラムでDフラグを変更しないでください。(FSET, FCLR命令を使用してください)	△
ISP (割り込みスタックポインタ)	05FFh	RAM領域内の値に設定変更が可能です (0800h以下の値)	○
UART1送受信モードレジスタ	15h	変更しないでください。	×
UART1転送速度レジスタ	—		
UART1送受信制御レジスタ0	00h		
UART1送受信制御レジスタ1	07h		
UART1送受信制御レジスタ2	22h		
UART1送信バッファレジスタ	—	データを書き込まないで下さい。	×
UART1受信バッファレジスタ	—	データを読み出さないで下さい。	×

○=変更可 △=変更可 (一部制限あり) ×=変更禁止

11.3 周辺機能に関する制限

UART1の送信および受信わりこみは、KD30モニタプログラムがホストPCとの通信に使用しますので、ユーザプログラムで使用しないでください。また以下の端子を使用しないでください。

TxD1 (32pin)、RxD1 (1pin)

11.4 KD30 の機能に関する制限

11.3.1 サンプリングモードとフリーランモードについて

●SamplingRun (サンプリング) モード

サンプリングモードでは、Go 実行時およびCome 実行時にユーザープログラムの実行状態を定期的に監視します。そのため、ブレークなどによるユーザープログラムの停止を検出することができます。通常のデバッグを行うときに選択してください。

●FreeRun (フリーラン) モード

フリーランモードでは、Go 実行時およびCome 実行時にユーザープログラムの実行状態を監視しません。そのため、ユーザープログラムのリアルタイム性は保証されませんが、ブレークなどによるユーザープログラムの停止を検出できません。したがって、ユーザープログラムが停止しても、KD30 はGo 実行およびCome 実行動作を停止しません。KD30 を停止させるには、STOP ボタンを押してください。ユーザープログラムのリアルタイム実行を行いたいときに選択してください。

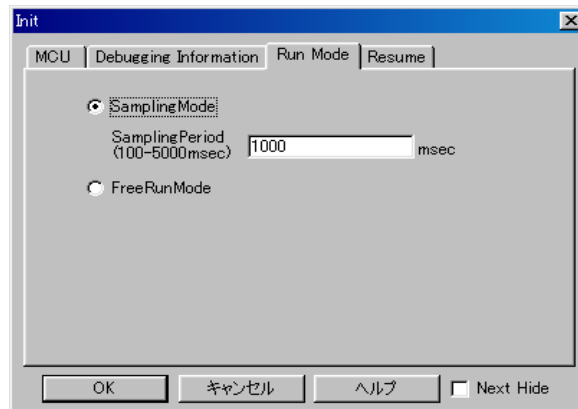


図11.2 Run Mode 設定画面 (KD30の初期画面)

11.3.2 ストップモード、ウェイトモードに関する制限事項

ユーザープログラムでストップモード、ウェイトモードを使用する場合、リモートデバッガはフリーランモードで起動するようにしてください。デバッグを行う場合は実行する前にあらかじめRAMウィンドウ、Cウォッチウィンドウ、ASMウィンドウを閉じてください。またストップモード、ウェイトモードを解除する処理部にブレークポイントを設定するなどして、ブレークポイントが止まるまで画面の操作をしないでください。

11.3.3 監視タイマのリアルタイム性について

ユーザープログラムで監視タイマを使用する場合、モニタプログラムは監視タイマの初期化を行っていますので、リアルタイム性が必要な場合は、KD30をフリーランモードで起動するようにしてください。またあらかじめRAMウィンドウ、Cウォッチウィンドウ、ASMウィンドウを閉じてください。

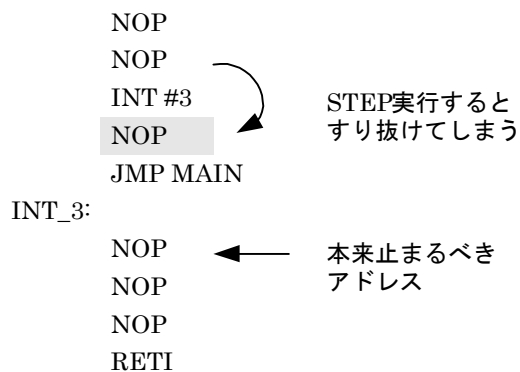
11.3.4 例外的なステップ実行について

ソフトウェア割り込みを発生させる命令（未定義命令、オーバーフロー、BRK命令、INT命令）の命令内部処理を連続してステップ実行することはできません。

例：INT 命令の場合

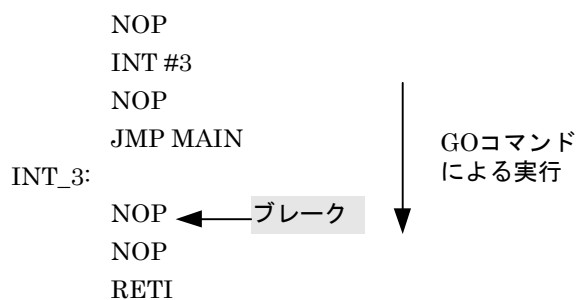
INT 命令から、INT 命令内部処理を連続してSTEP 実行はできません。

<例>



INT 命令を用いたプログラムのデバッグは、INT 命令内部処理にソフトウェアブレイクを設定し、GO コマンドと共に使用して下さい。

<例>

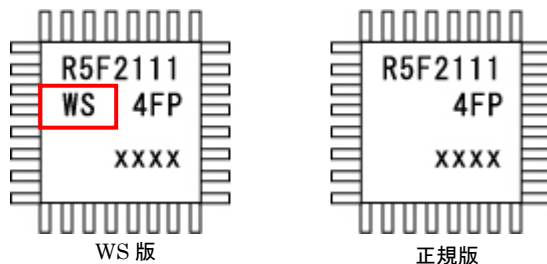


12. WS 版における相違点

OAKS8に搭載されておりますマイクロコンピュータ R5F21114FP には、WS（ワークサンプル）版（初期ロットのみ）と正規版があります。WS版と正規版は、機能的には同じものです。相違点は以下のとおりです。

12.1 WS 版と正規版の主な相違点

- WS版は、チップ表面に“WS”と印字があります。正規版には“WS”の印字がありません。



- 半導体工場での出荷時の品質テストレベルが違います。WS版を量産品に組み込む事はできません。
- 内部ブートエリアにあらかじめ書き込まれているブートプログラムのバージョンが違います。
WS版→Ver 0.9 正規版→Ver 1.0

ブートプログラムのバージョンが違う事で、リモートデバッグKD30の起動時の動作、メモリマップ、設定するIDコードに違いがあります。

	WS版（ブートプログラム Ver0.9）	正規版（ブートプログラム Ver1.0）
KD30起動時の動作	現在のフラッシュメモリの内容を全て消去してから、モニタプログラムを書き込み、モニタプログラムへ実行を移します。	現在のフラッシュメモリに設定されているIDコードがすべて“00h”か“FFh”のときのみ、フラッシュメモリの内容を全て消去してから、モニタプログラムを書き込みモニタプログラムへ実行を移します。 IDコードが“00h”か“FFh”以外の場合は、何もしません。 (ホストPCでは、モニタプログラム書き込みのプログラレスパーが止まった状態のままとなります)
KD30が使用する領域	RAM：700H～7FFH（400H～6FFHまで使用可） ROM：C000H～0CBFFH（C000H～FFFFHまで使用可）	RAM：制限なし ROM：C000H～0C7FFH（C800H～FFFFHまで使用可）
KD30起動の際に設定するIDコード	IDコードのID3, 4, 6に“00h”を設定します。ユーザプログラムをダウンロードしたあとは、IDコードは以下になります。 xx xx 00 00 xx 00 xx (xはユーザが設定した値) KD30起動後、ユーザプログラムをダウンロードしないで終了した場合は、以下になります。 FF FF 00 00 FF 00 FF	すべてのIDコードに“FFh”を設定します。KD30起動後は、ユーザプログラムをダウンロードしてもしなくてもIDコードはAll FFhになります。

12.2 ブートプログラム Ver0.9 の場合のメモリマップ

マイクロコンピュータがWS版の場合、ブートプログラムはVer0.9となります。KD30使用時にユーザが利用できる涼気は以下のようになります。

RAM : 400h~6FFh (700h~7FFhまでモニタプログラムが使用)

ROM : CC00~FFFFh (C000~CBFFhとベクタ領域、IDコードの一部をモニタプログラムが使用)

C000h~CBFFhの領域にプログラムを書き込まないで下さい。KD30は、ユーザプログラムがモニタプログラムと重なった場合、モニタプログラムと重なった領域は書き込みません。また、この場合KD30はメッセージ等を出しませんのでご注意ください。

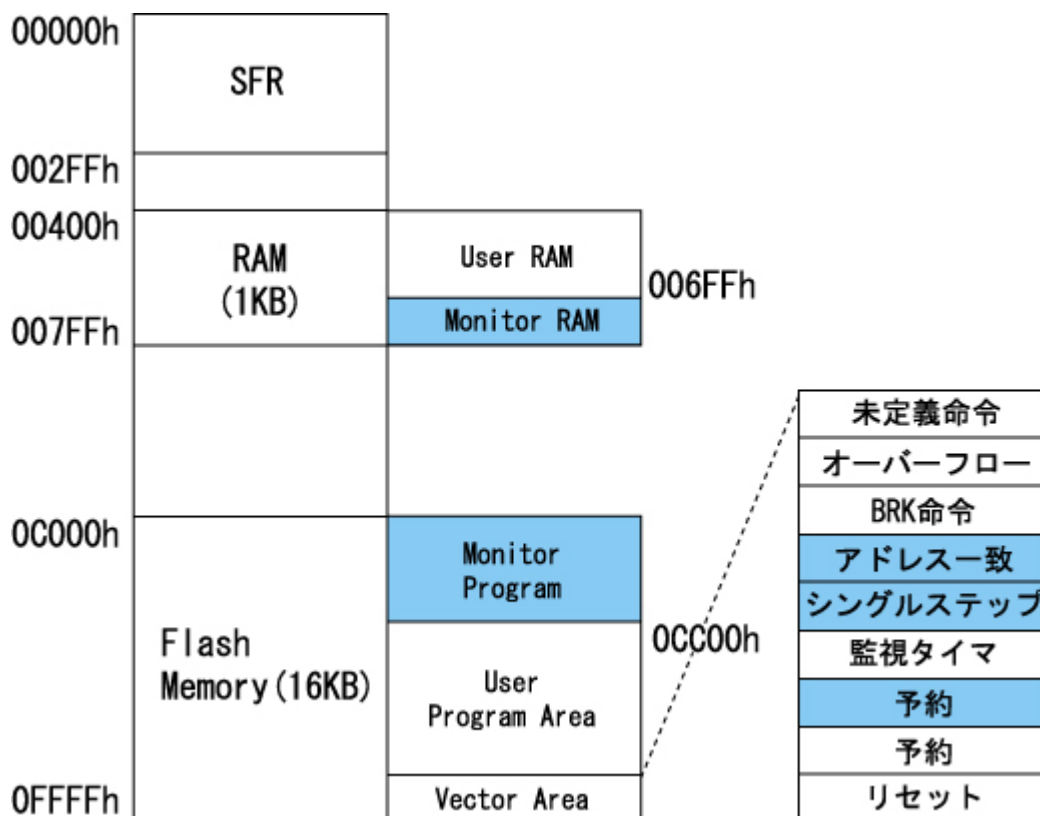


図12.1 KD30使用時のメモリマップ (ブートバージョンが0.9の場合)

12.3 ブートプログラム Ver0.9 の場合に KD30 が設定する ID 番号

マイクロコンピュータがWS版の場合、ブートプログラムはVer0.9となります。ブートプログラムがVer0.9の場合モニタプログラムでは以下のベクタアドレスを使用しており、またIDコードも00hに設定しています。
 KD30起動後は必ずID3, 4, 6は00hとなっていますので、FlashStartでIDコードの照合を行う際にはご注意ください。
 混乱を避けるために、プログラム内とIDオプションの両方で設定する場合は、同じIDコードを設定することをお勧めします。

は、モニタプログラムが使用、設定するベクタ領域です。

OFFDCh	未定義命令ベクタ	OFFE8h	アドレス一致ベクタ	OFFF4h	予約
	未定義命令ベクタ		アドレス一致ベクタ		予約
OFFDFh	ID1	OFFEBh	ID3=00h	OFFF7h	ID6=00h
OFFE0h	オーバーフローベクタ	OFFECh	シングルステップベクタ	OFFF8h	予約
	オーバーフローベクタ		シングルステップベクタ		予約
OFFE3h	ID2	OFFEFh	ID4=00h	OFFFBh	ID7
OFFE4h	BRK命令ベクタ	OFFF0h	監視タイマベクタ	OFFFCh	リセットベクタ
	BRK命令ベクタ		監視タイマベクタ		リセットベクタ
		OFFF3h	ID5	OFFFh	

図12.2 KD30が設定するIDコード（ブートバージョン0.9の場合）

12.3.1 KD30 でユーザプログラムをダウンロードした場合設定される ID

★KD30では、リロケータブルファイル（.x30）をダウンロード（フラッシュメモリに書き込む）します。

- ① ユーザプログラムでIDコードを設定していた場合
 フラッシュメモリに書き込まれるIDコードは以下のようになります。
 (IDオプションは機械語ファイルのみに反映されますので、IDオプションでのみ指定した場合は②に相当します)

ID1	ユーザが設定した値
ID2	ユーザが設定した値
ID3	00h
ID4	00h
ID5	ユーザが設定した値
ID6	00h
ID7	ユーザが設定した値

- ② LMCのIDオプションでのみIDコードを設定してるか、全く指定していない場合
 フラッシュメモリに書き込まれるIDコードは以下のようになります

ID1	FFh
ID2	FFh
ID3	00h
ID4	00h
ID5	FFh
ID6	00h
ID7	FFh

12.4 ブートプログラムのバージョン確認方法

マイクロコンピュータに書き込まれているブートプログラムのバージョンは、Flash Starterで確認できます。

- ①OAKS8ボードとホストPCを接続し、ボードの電源を投入してから Flash Starterを起動します。各メニューウィンドウを表示するためには、ファイル名（書き込まなくても）、現在のIDコードを入力する必要があります。

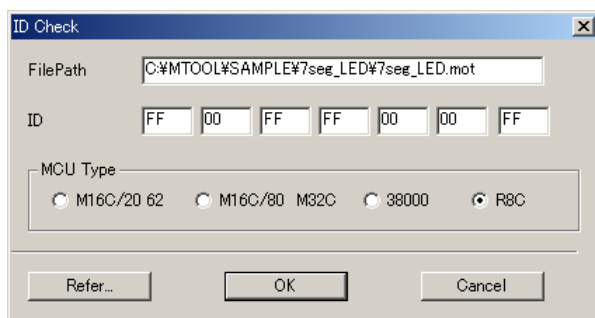


図12.3 Flash Starterの起動ウィンドウ

メニューウィンドウが開いたら、“Version” ボタンをクリックしてください。マイクロコンピュータのブートプログラムのバージョンが表示されます。

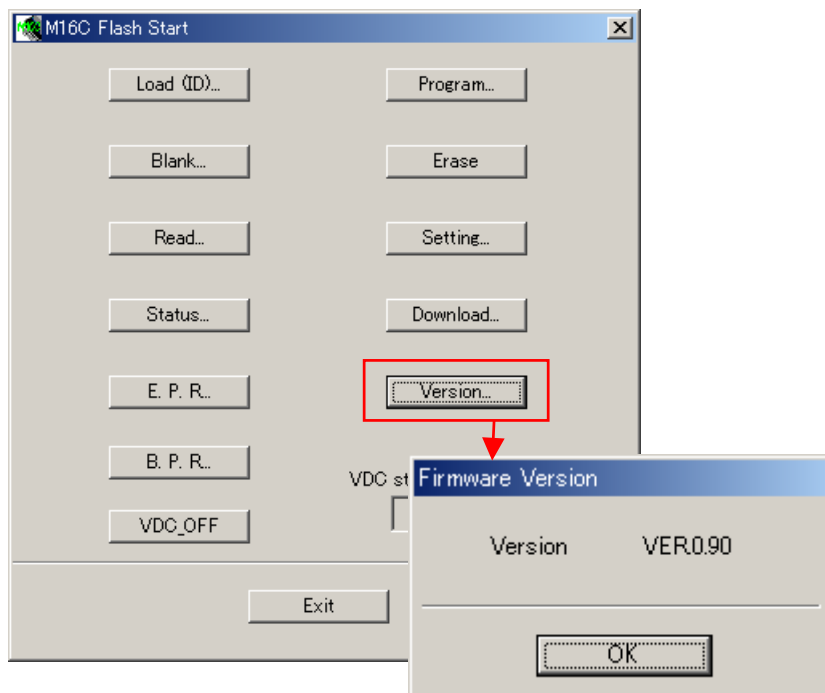


図12.4 ブートバージョンの確認（WS版の場合）

12.5 WS版でサンプルプログラムを動作させる場合の修正事項

本キットで提供しているサンプルプログラムでは、正規版のメモリマップを元にして、各セクションにアドレスを設定しています。お手持ちのキットに搭載のマイクロコンピュータがWS版の場合は、下記のとおり修正して、再コンパイルしてください。

12.5.1 プログラム開始アドレスの修正

NCRT0.A30ファイルのrom_topを0C800Hから0CC00Hに修正してください。

```
***** ;
; C_startup for R8C/Tiny
*****

ram_top      .equ      400h
rom_top      .equ      0CC00H
```

リスト12.1 NCRT0.A30 ROM先頭番地の設定

12.5.2 スタックポインタ初期値の修正

NCRT0.A30ファイルのistack_topを7ffh+1から6ffh+1に修正してください。

```
***** ;
; C_startup for R8C/Tiny
*****

ram_top      .equ      400h
rom_top      .equ      0CC00H

istack_top   .equ      6ffh+1      ; bottom of R8C/11 RAM area
```

リスト12.2 NCRT0.A30スタック領域開始番地の設定

13. 保証とサポート

13.1 保証について

製品の出荷には万全を期しておりますが、欠品、破損、初期不良などがありましたら弊社までご連絡ください。

お問い合わせ先

オークス電子株式会社

〒101-0025

東京都千代田区神田佐久間町3丁目21番地

(第一千代田ビル3F)

TEL 03-3863-1121

FAX 03-3863-1130

13.2 技術サポートについて

E-MAILにて技術的なご質問を承っております。なおご質問の内容によっては回答に時間がかかる場合がありますのでご了承ください。又弊社サイトでもQ&Aを掲載しております。

E-MAIL: oaks8support@oaks-ele.com

URL : <http://www.oaks-ele.com>

改訂記録

Version	発行日	ページ	改訂内容
0.98	2004.2.1	-	初版発行
1.00	2004.7.15	32	「図7.3 IDオプションの設定」を修正
		-	その他の図、文章の改訂
2.00	2004.9.17	-	WS版から正規版チップへの移行
	2004.10.4	17	-gオプションを“その他”欄で設定する→“デバッグ”カテゴリで設定する
		33	IDコードがAll “FFh” のときのみモニタプログラムを書き込む→ IDコードがAll “00h” か “FFh” のときのみ…に修正
		3	「はじめに」に正規版とWS版の違いを追加
		40	「11章 制限事項」を追加
			「12章 WS版との相違点」を追加

OAKS8-FullKit クイックツアー Rev 2.00
2004年9月発行

編集 オークス電子株式会社
発行 オークス電子株式会社
禁無断転載

本説明書の一部又は全部を、当社に断りなく、いかなる形でも転載又は複製することを堅くお断りします。