

ビルドシステム

ビルドシステムとソースツリーの説明

2008-05-30

任天堂株式会社発行

このドキュメントの内容は、機密情報であるため、厳重な取り扱い、管理を行ってください。

目次

1	はじめに	4
2	ビルドシステム	4
2.1	環境変数	4
2.2	ビルドツール	4
2.2.1	Makefileの記述	5
2.2.2	プラットフォームの設定	5
2.2.3	TWL用コード生成の選択	5
2.2.4	コンパイルターゲットの設定	6
2.2.5	その他のビルドスイッチ	6
2.2.6	ターゲット	6
3	ソースツリー	7
3.1	インクルード	7
3.2	ライブラリ	8
3.2.1	ライブラリファイルの命名規則	8
3.3	ビルドツリー	9
3.4	ライブラリとデモのサブディレクトリ構造	10
3.5	ビルドに必要なファイル	11
3.5.1	commondefsファイル	11
3.5.2	modulerrulesファイル	11
3.5.3	nnslibdefsファイル	11
3.5.4	commondefs.cctype.CWファイル	11

表

表 2-1	TWLSDK_PLATFORMに設定可能な値	5
表 2-2	TWL_ARCHGENIに設定可能な値	5
表 2-3	利用可能なビルドスイッチ	6
表 2-4	利用可能なターゲット	6
表 3-1	ライブラリファイルの命名規則	8

図

図 3-1	ソースツリー第1階層のディレクトリ	7
図 3-2	インクルードディレクトリ構造	7
図 3-3	ライブラリディレクトリ構造	8
図 3-4	ビルドディレクトリ構造	9
図 3-5	ライブラリとデモの基本ディレクトリ構造	10

改訂履歴

改訂日	改訂内容
2008-05-30	<ul style="list-style-type: none"> •NITRO-System の名称変更による修正 (NITRO-System から TWL-System に変更)。 •「2.2 ビルドツール」を改訂。 •「3.2.1 ライブラリファイルの命名規約」を改訂。 •図を全面的に改訂。
2008-04-08	<ul style="list-style-type: none"> •クイックスタートを QuickStart.pdf に分離。 •TWL-SDK に対応。
2006-05-29	•NITRO-SDK のサウンドパッチに関する記述を削除。
2004-10-12	<ul style="list-style-type: none"> •P.10 ビルドに必要なファイルに、nnslibdefs と commondefs.cctype.CW の説明を追加。 •SDK の表記に合わせて、変数と言う語をマクロスイッチに変更。 •文章や図中の TEG という表記を TS に変更。
2004-08-10	図 3-4 ビルドディレクトリ構造を修正。
2004-05-28	P.8 「インターフェース」と言う表記を「インクルード」に変更。
2004-04-12	•誤字の修正。
2004-04-08	<ul style="list-style-type: none"> •NITRO-SDK, IS-NITRO-CHARACTER 等の用語を統一。 •商標表記を修正。
2004-04-02	P.4 NITRO-SDK サウンドドライバのインストールに関する項目を追加。 P.6 ビルドツール (commondefs と modulerules のインクルードについて) の説明を補足。
2004-02-20	サブプロセッサ (ARM7) 側のビルドに関する記述を削除。
2004-02-20	図 3-5、図 3-6 に depend ディレクトリを追加。
2004-02-13	サブプロセッサのソースとヘッダファイルの格納場所が変更された為、ソースツリーの説明を大幅に修正。

1 はじめに

このドキュメントでは、TWL-System のビルドシステムとソースツリーについて説明しています。なお、TWL-System ライブラリのビルドシステムは、TWL-SDK のビルドシステム上に構築されておりますので、TWL-SDK のドキュメントも合わせてご覧ください。

2 ビルドシステム

2.1 環境変数

TWL-System ライブラリを使ったプログラムをビルドする場合には、環境変数 `TWLSYSTEM_ROOT` に、TWL-System のルートディレクトリ (`TwlSystem`) の絶対パスが設定されている必要があります。環境変数 `TWLSYSTEM_ROOT` が指定されていない場合には、`C:\¥TwlSystem` が設定されたものとして扱われます。以後このディレクトリのことを `$TwlSystem` と表記します。

2.2 ビルドツール

TWL-System では、TWL-System ライブラリを利用するアプリケーション用の `Makefile` を容易に記述できるようにするために、よく使われる手順についての記述をまとめたファイルを以下のディレクトリに用意しています。

- ディレクトリ: `$TwlSystem/build/buildtools/`
- マクロスイッチなどの定義ファイル: `commondefs`
- コンパイル手順定義ファイル: `modulerules`

TWL-System ライブラリを利用したアプリケーションを作成される場合には、この2つのファイルを `Makefile` 中にインクルードして使います。これらのファイルの使用方法は、TWL-System ライブラリのサンプルプログラム等のコンパイルに使用している `Makefile` をご参考ください。

なお、TWL-System のビルドシステムは、TWL-SDK のビルドシステムの上に構築されています。これらの設定ファイルの中では、TWL-SDK の `commondefs` ファイルと `modulerules` ファイルをそれぞれインクルードしており、TWL-SDK の設定に TWL-System ライブラリ固有の設定を付け加えると言う形となっています。よって、TWL-System 版の `commondefs` ファイルと `modulerules` ファイルをインクルードする場合には、TWL-SDK の `commondefs` ファイルと `modulerules` ファイルをインクルードする必要はありません。

2.2.1 Makefileの記述

TWL-System を用いた Makefile の記述やビルドの仕方は、TWL-SDK のみを利用した開発の場合とほとんど同じとなっています。TWL-SDK の Makefile との唯一の違いは、`commondefs` ファイルと `modulerules` ファイルをインクルードしている部分のみとなります(環境変数の名前のみが異なります)。

- TWL-SDK の Makefile でのインクルード

```
include $(TWLSDK_ROOT)/build/buildtools/commondefs
include $(TWLSDK_ROOT)/build/buildtools/modulerules
```
- TWL-System の Makefile でのインクルード

```
include $(TWLSYSTEM_ROOT)/build/buildtools/commondefs
include $(TWLSYSTEM_ROOT)/build/buildtools/modulerules
```

2.2.2 プラットフォームの設定

TWL-System ライブラリを利用したアプリケーションをビルドする場合には、ビルド対象とするプラットフォームを設定する必要があります。ビルド対象とするプラットフォームは、環境変数 `TWLSDK_PLATFORM` に設定します。

この環境変数 `TWLSDK_PLATFORM` に設定できる値を表 2-1 に示します。

表 2-1 TWLSDK_PLATFORM に設定可能な値

環境変数の値	生成されるコード
TWL	TWL 用のコードを生成します。
NITRO	NITRO 用のコードを生成します。
ALL	TWL と NITRO 用の両方を生成します。
TWL NITRO	

なお、環境変数 `TWLSDK_PLATFORM` にはデフォルトの値が設定されていないので、必ず設定する必要があります。

2.2.3 TWL用コード生成の選択

プラットフォームの設定により TWL 用のコードを生成する場合には、TWL 専用のコードを生成するか、または TWL/NITRO ハイブリッド用のコードを生成するかを選択することができます。生成するコードを選択するには TWL-SDK のビルドスイッチ `TWL_ARCHGEN` を設定します。

このビルドスイッチ `TWL_ARCHGEN` に設定できる値を表 2-2 に示します。

表 2-2 TWL_ARCHGEN に設定可能な値

TWL_ARCHGEN の値	生成されるコード
LIMITED	TWL 専用コードを生成します。
HYBRID	TWL/NITRO ハイブリッド用コードを生成します。
ALL	TWL 専用コードとハイブリッド用コードの両方を生成します。
LIMITED HYBRID	

なお、ビルドスイッチ `TWL_ARCHGEN` を明示的に設定しなかった場合には、TWL/NITRO ハイブリッド用のコードを生成します。

2.2.4 コンパイルターゲットの設定

TWL-SystemでのビルドでもTWL-SDKと同様に、DEBUG版、RELEASE版、FINALROM版の3つコンパイルターゲットを選択することができます。コンパイルターゲットの指定するには、表 2-3 に示す3つのビルドコンパイルのいずれかひとつに適切な値(TRUEなど)を設定します。コンパイルターゲットを明示的に指定しなかった場合は、RELEASE版のコードが生成されます。

表 2-3 利用可能なビルドスイッチ

コンパイルターゲット	生成されるコード
% make TWL_DEBUG=TRUE	DEBUG 版のコードを生成します。
% make TWL_RELEASE=TRUE	RELEASE 版のコードを生成します。
% make TWL_FINALROM=TRUE	FINALROM 版のコードを生成します。

2.2.5 その他のビルドスイッチ

TWL-System ライブラリのビルドシステムは、TWL-SDK のビルドシステム上に構築されていますので、TWL-SDK のビルドスイッチが使用可能です。

TWL-SDK のビルドシステムには、これまでの説明に出てきましたビルドスイッチ以外にも多くのビルドスイッチが用意されています。TWL-SDK のビルドスイッチについての詳しい情報は、下記の TWL-SDK のドキュメントをご覧ください。

[\\$TwlSDK/docs/SDKRules/Rule-Defines.html](#)

2.2.6 ターゲット

TWL-System ライブラリでは、TWL-SDK で用意されている幾つかのターゲットを利用する事ができます。下表に利用可能なターゲットを示します。

表 2-4 利用可能なターゲット

コマンド	処理
% make build	コンパイルを開始し、最終ターゲットを作成します。
% make install	make build によって作成されたファイルを他のディレクトリへインストール(コピー)します。
% make run	IS-NITRO-EMULATOR が使用可能な環境の場合、make build で生成したターゲットファイルの実行を始めます。
% make full	make build 各コンパイルターゲット毎全てのバージョンのファイルを生成します。
% make clean	make build によって生成されたファイルを削除します。
% make clobber	make build によって生成されたファイルを完全に削除します。

3 ソースツリー

図 3-1 にTWL-Systemのソースツリー第1階層にあるディレクトリを示します。

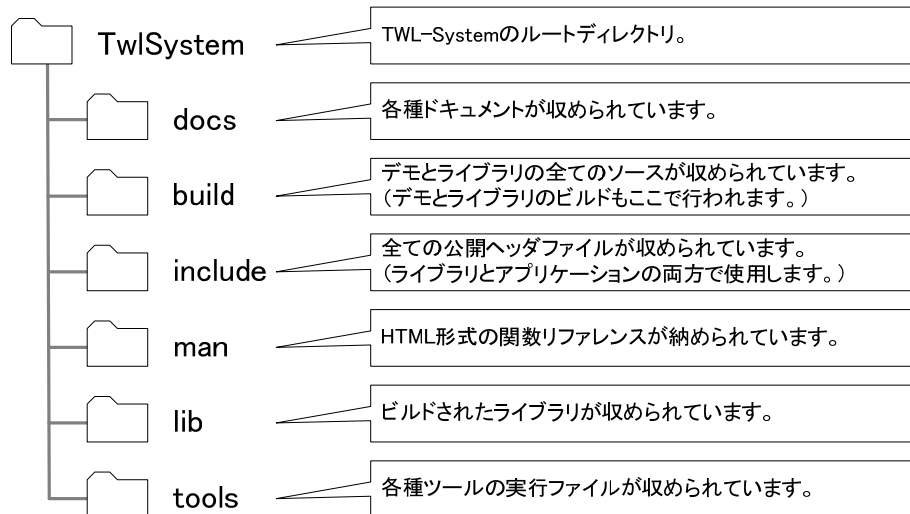


図 3-1 ソースツリー第1階層のディレクトリ

TwlSystem ディレクトリの中には、6個のディレクトリがあります。これら6個のディレクトリの中で、アプリケーションのビルドに関係するディレクトリの構造について説明します。

3.1 インクルード

図 3-2 に、インクルードディレクトリの構造を示します。

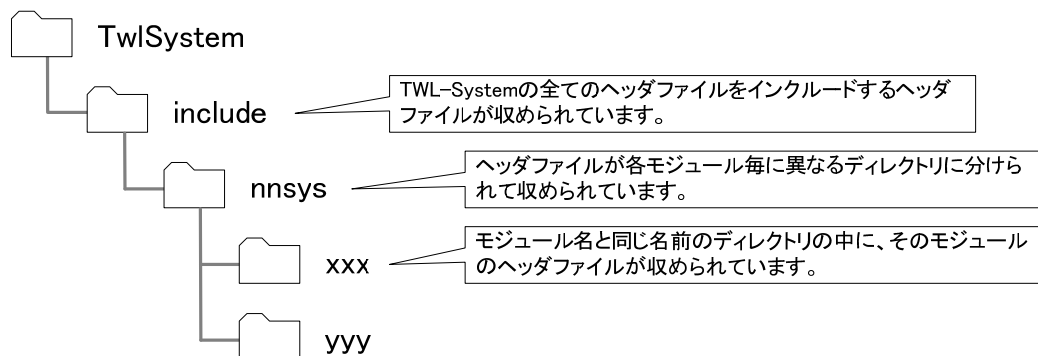


図 3-2 インクルードディレクトリ構造

TWL-System ライブラリのすべてのシステムインクルードパスは、`$TwlSystem/include` からの相対パスとなっています。`$TwlSystem/include` には、TWL-System ライブラリのすべてのヘッダファイルをインクルードする為のヘッダファイルである `nnsys.h` が収められています。このファイルをインクルードする場合には、以下の様に指定します。

```
#include <nnsys.h> // ヘッダファイルをすべてインクルード。
```

各モジュールのヘッダは、\$TwlSystem/include の中にある nnsys ディレクトリ内に、モジュール毎に異なるディレクトリに分けられて格納されています。アプリケーションでモジュールを特定したヘッダファイル(Foundation ライブラリ、NITRO-Composer など)をインクルードしたい場合は、以下の様に指定します。

```
#include <nnsys/fnd.h> // Foundationライブラリのヘッダファイルをすべてインクルード。
#include <nnsys/snd.h> // NITRO-Composerのヘッダファイルをすべてインクルード。
```

3.2 ライブラリ

図 3-3 に、ライブラリディレクトリの構造を示します。

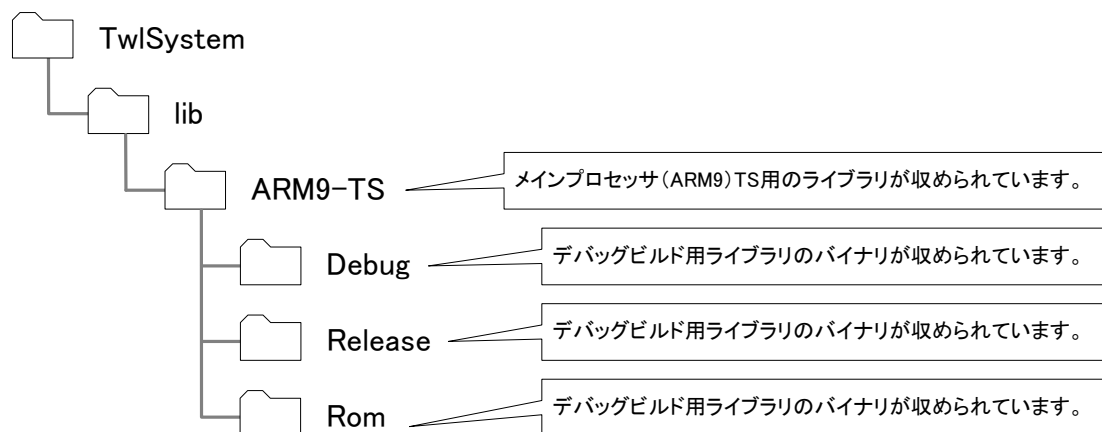


図 3-3 ライブラリディレクトリ構造

TWL-System ライブラリのバイナリファイルは、全て\$TwlSystem/lib ディレクトリに収められています。TWL-System ライブラリのビルドシステムでは、指定されたビルドスイッチに従って使用するライブラリを切り替えます。必要なライブラリは全てリンクに渡されますので、開発者はどのライブラリをリンクすべきかを意識する必要はありません。

3.2.1 ライブラリファイルの命名規則

TWL-System ライブラリの名前は、ライブラリを示す接頭辞 “lib” の後ろに、TWL-System に属する事を示す語 “nns” が置かれ、その後にアルファベット2～3文字のモジュール名 (ライブラリ名) が続いたものが基本となります。

TWL で使用可能なライブラリの場合には、モジュール名の後ろにライブラリのプラットフォームを示す語が追加されます。また THUMB モードでビルドされたライブラリには、最後に “.thumb” という語が付きます。

表 3-1 ライブラリファイルの命名規則

ライブラリ名	ライブラリの種類
libnns + モジュール名.a	NITRO 用ライブラリ (ARM モード)
libnns + モジュール名.thumb.a	NITRO 用ライブラリ (THUMB モード)
libnns + モジュール名.TWL.HYB.a	TWL, NITRO 両用ライブラリ (ARM モード)
libnns + モジュール名.TWL.HYB.thumb.a	TWL, NITRO 両用ライブラリ (THUMB モード)
libnns + モジュール名.TWL.LTD.a	TWL 用ライブラリ (ARM モード)
libnns + モジュール名.TWL.LTD.thumb.a	TWL 用ライブラリ (THUMB モード)

以下に、実際のライブラリ名の例を示します。

```
libnnsfnd.thumb.a      // NITRO用のFoundationライブラリ（THUMBモード）。  
libnns2d.TWL.HYB.a     // TWL, NITRO両用の2Dライブラリ（ARMモード）。  
libnns3d.TWL.LTD.thumb.a // TWL用の3Dライブラリ（THUMBモード）。
```

3.3 ビルドツリー

図 3-4 に、ビルドディレクトリの構造を示します。buildディレクトリ以下には、ライブラリやデモプログラムのソースコードが格納されており、ここで、ライブラリやデモプログラムがビルドされます。

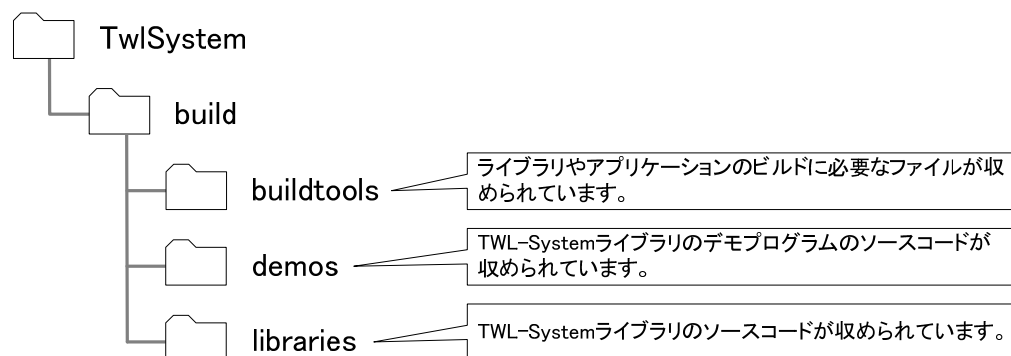


図 3-4 ビルドディレクトリ構造

ライブラリのソースコードは、libraries ディレクトリ内にモジュール毎に異なるディレクトリに分けられて格納されています。buildtools ディレクトリには、ライブラリやアプリケーションソフトのビルドに使用される Makefile にインクルードされる、commondefs と modulerules ファイルが収められています。

3.4 ライブラリとデモのサブディレクトリ構造

ライブラリとデモプログラム各モジュールは、基本的には 図 3-5 に示すようなディレクトリ構造となっています。

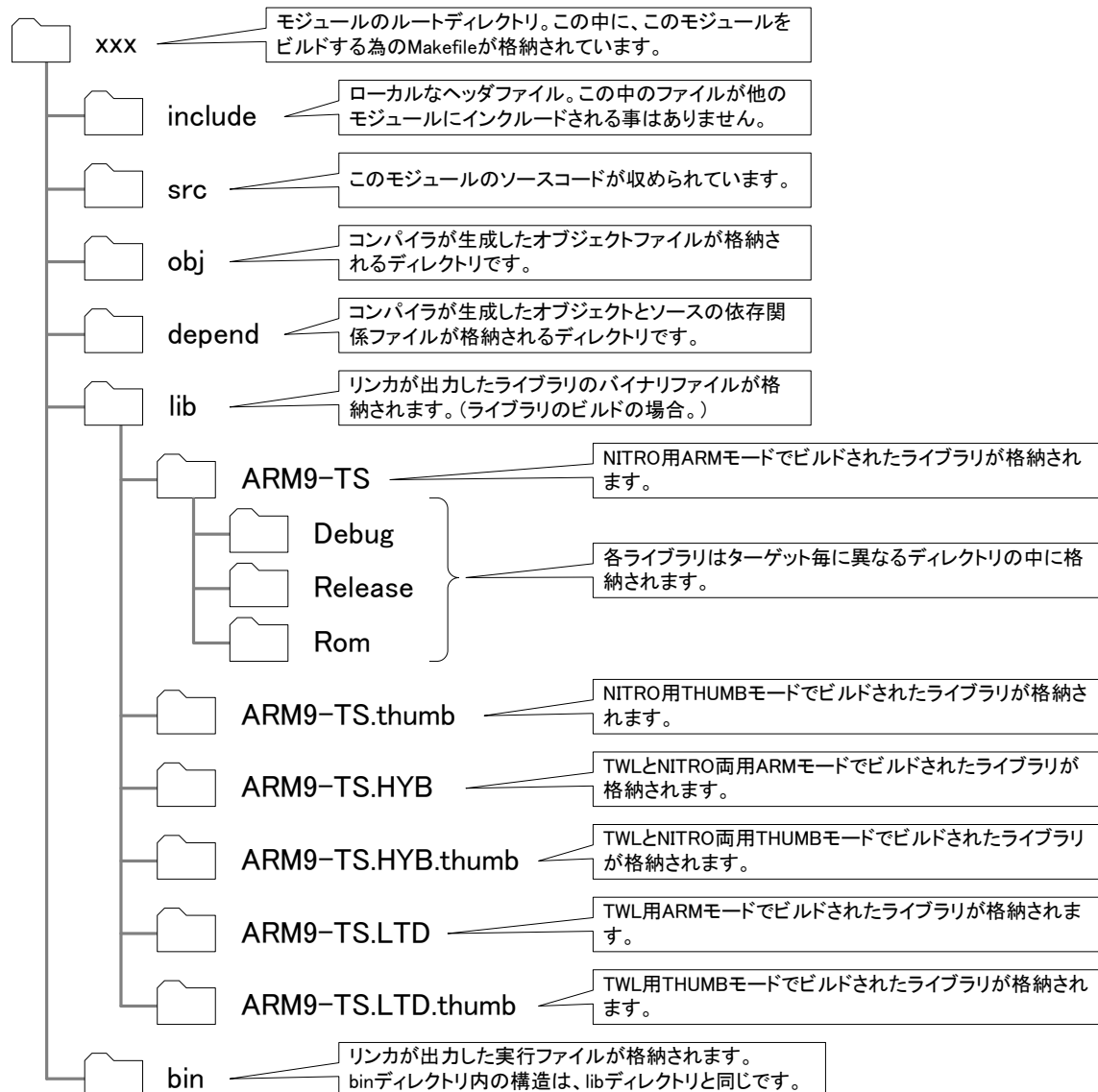


図 3-5 ライブラリとデモの基本ディレクトリ構造

各モジュールのルートディレクトリには、そのモジュールをビルドする為の Makefile が置かれます。この Makefile は、依存関係の生成とコンパイラとリンカの起動を行うために、\$TwlSystem/build/buildtools の中に格納されている commondefs ファイルと modulerules ファイルを使用します。

各モジュール名を持ったディレクトリ内にあるローカルな include ディレクトリは、モジュール間で共有しない専用のヘッダファイルのために存在します。

3.5 ビルドに必要なファイル

TWL-System のライブラリや、TWL-System ライブラリを使ったアプリケーションをビルドするためには、\$TwlSystem/build/buildtools/ディレクトリに格納されている下記のファイルを用います。これらのファイルは、Makefile の中からインクルードされます。

3.5.1 commondefsファイル

commondefs ファイルでは、TWL-System ライブラリのビルドに必要なマクロスイッチが定義されています。TWL-System ライブラリの commondefs ファイルは、内部で TWL-SDK の commondefs ファイルをインクルードしています。このファイルでは、TWL-SDK の commondefs ファイルで行われている設定に加え、TWL-System ライブラリに関するマクロスイッチの設定が行われています。

3.5.2 modulerulesファイル

現在、TWL-System ライブラリの modulerules ファイルは、内部で TWL-SDK の modulerules ファイルをインクルードしているのみとなっています。しかし将来、なんらかの設定が付け加えられる可能性がありますので、TWL-System ライブラリを利用される場合は、TWL-SDK の modulerules ファイルを直接使用せずに、TWL-System の modulerules ファイルをお使い下さい。

3.5.3 nnslibdefsファイル

nnslibdefs ファイルは、TWL-SDK の commondefs ファイルからインクルードされます。このファイルでは、TWL-System ライブラリのインクルードパスとライブラリのパスの設定、及びリンカに渡されるライブラリの設定を行っています。

3.5.4 commondefs.cctype.CWファイル

commondefs.cctype.CW ファイルは、TWL-System の commondefs ファイルからインクルードされます。このファイルでは、TWL-System ライブラリで使用されているマクロスイッチの設定が行われています。

© 2004-2008 Nintendo

任天堂株式会社の許諾を得ることなく、本書に記載されている内容の一部あるいは全部を無断で複製・複写・転写・頒布・貸与することを禁じます。