SEGGER emWin Graphic Library (emWinGraphics) 1.0

特長

- このコンポーネントは、emWin 8051 Graphic Library を PSoC3 に、フル機能の emWin Graphic Library V5.02 を PSoC 5 に取り込みます。
- このライブラリーは、Keil_PK51、GCC、Keil MDK、および Keil RVDS 各ツールチェーンで使用できます。
- Graphics LCD Interface コンポーネントと Graphics LCD Controller コンポーネント用のドライバが用意されています。

概要説明

emWin は組み込みグラフィック ライブラリであり、グラフィカル表示とともに動作するアプリケーションに対して、プロセッサおよび LCD コントローラと独立した効率的な GUI を提供するために設計されたグラフィカル ユーザーインターフェイス (GUI) です。これは、シングルタスクおよびマルチタスク環境に対応しています。SEGGER Microcontroller によって開発された emWin は、組み込み業界で非常に有名です。サイプレスは SEGGER から emWin ライブラリのライセンスを供与されており、お客様にフル機能のグラフィック ライブラリを無料で提供します。

emWinGraphics を使用する場合

このコンポーネントは、SEGGER emWin グラフィック ライブラリへのアクセスと機能を提供します。 emWinGraphics コンポーネントは、PSoC 3 および PSoC 5 のターゲット ハードウェア用の完全で使い易い ソフトウェア パッケージです。このパッケージは、ライブラリ形式で提供された emWin GUI、必須のヘッダーファイル、および PSoC 実装に固有のソース ファイルで

構成されています。emWin Graphics Library コンポーネントは、グラフィカル LCD とともに動作するアプリケーション用に、高速で効率的な GUI 開発を可能にします。

はじめに

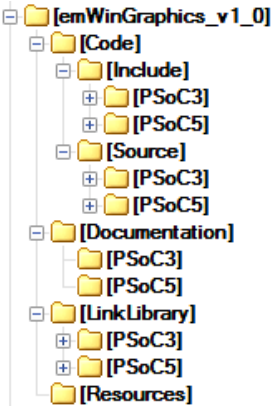
セットアップ

emWinGraphics コンポーネントは、zipファイルで供給されます。コンポーネントの最新バージョンは、次の場所から入手できます:

http://www.cypress.com/go/comp_emWin

この zip ファイルを、ディレクトリ構造を維持したまま、任意のフォルダに解凍します。 解凍後のファイルが読み取り 専用になっていないことを確認します。zip ファイルを解凍した後のインストールのディレクトリ構造は、図 1 に示す ようなものになります。

図 1. ディレクトリ構造



Resources を除く、主要ディレクトリはそれぞれ PSoC 3 と PSoC 5 のサブディレクトリに分かれます。これは、 2 つのパーツ ファミリーでは実装が異なるためです。PSoC 3 は限定されたライブラリ バージョン (emWin 8051) をサポートしていますが、PSoC 5 は標準 emWin ライブラリのフル機能バージョンをサポートしています。各ライ ブラリ バージョンによって提供される機能の詳細については、*Documentation\PSoC3* および *Documentation\PSoC5* ディレクト

リにある SEGGER emWin ドキュメンテーションに 記載されています。 PSoC 3 と PSoC 5 の機能が異なる領域の詳細については、本書の機能の説明セクションを参照してください。

次の表に、主要 emWinGraphics ディレクトリの内容を示します。

ディレクトリ	内容
Code\Include	GUI ヘッダー ファイル
Code\Source	プロジェクトへの追加が必要なソース ファイル
ドキュメント	emWin User Guides (ユーザ ガイド)
LinkLibrary	サポートされる各ツールチェーンの emWin ライブラリ
リソース	SEGGER からのビットマップを使うための emWin サンプル ファイルと Windows プログラム

emWinGraphics Library の使用

emWinGraphics ライブラリを使用するには、次の手順に従います。すべての手順がその後のセクションでさらに説明されています。

1. emWinGraphics Library をあなたの PSoC Creator プロジェクトに統合します。
2. グラフィックス LCD パネルと通信するためのコンポーネントをプロジェクトに追加します。emWinGraphics ライブラリはソフトウェア ライブラリです。パネルとの通信は、パネルに適したハードウェア コンポーネントを使用して行われます。Cypress コンポーネント ライブラリには、LCD パネルとのインターフェイスとなる 2 つのコンポーネント: Graphics LCD Interface または Graphics LCD Controller が含まれています。
3. プロジェクトにタッチスクリーン コンポーネントを追加します (オプション)。抵抗膜式タッチパネルなどのタッチスクリーン コンポーネントが、emWinGraphics ライブラリで使用できます。
4. 付属のサンプル コードを使用してコンパイル、リンクおよびテストします。emWinGraphics ライブラリには、 シングルおよびマルチタスク環境用のサンプル コードが付属しています。

これらのサンプルを使って、ライブラリの使用方法を学習できます。こられのサンプルは、Resources ディレクトリにあります。

5. マルチタスキングの場合は、アプリケーションはライブラリをご使用のオペレーション システム (OS) に適合させます。複数のタスクが同時にディスプレイにアクセスする場合は、OS のインターフェイス ルーチンを定義する必要があります。詳細とサンプル アプリケーションについては、第 13 章: 「実行モデル: シングルタスク/マルチタスク」(*emWin User Manual*) を参照してください。
6. emWin を使用して独自のアプリケーションを書きます。この時点で、emWin を使用してプログラミングを開始する準備が完了している必要があります。ライブラリが提供する各機能の詳細については、*emWin User Manual* のリファレンスの章を参照してください。

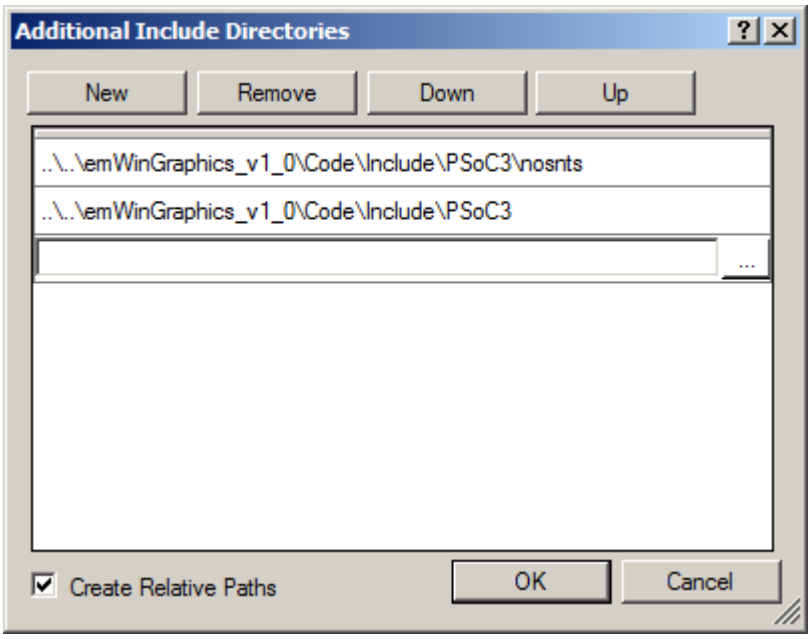
emWinGraphics Library を PSoC Creator に統合する

emWinGraphics Library をあなたのプロジェクトに統合するには、次の手順に従います。この例では、Keil_PK51 ツールチェーンを使用します。その他のツールチェーンも、同様の方法がサポートされています。

1. 初めに、必要なライブラリを決定します。この決定は、RTOS (os)、タッチスクリーン (ts)、またはその両方を使用するかどうかによります。emWin ライブラリは、サポートされるオプションに応じて指名されます。この例では、ライブラリ名は emWinnosnts.lib です。これは、OS (nos) とタッチスクリーン (nts) サポートを持たない ライブラリです。
2. 必要なインクルード ファイルを追加します。emWin include ディレクトリには、アプリケーション全体に対する全般的なインクルード ファイルが格納され、特定のインクルード ファイルは選択したオプションに基づいてサブディレクトリに入ります。Compiler オプションの **Additional Include Directories** フィールドには、全般的なインクルード ファイルとライブラリ固有のインクルード ファイルのパスを指し示すエントリを入れます。

   メニューから **Project (プロジェクト) > Build Settings (ビルド設定) > Compiler (コンパイラ) > General (全般) > Additional Include Directories (追加のインクルード ディレクトリ)** を選択して、インクルード パスを指定します (図 2 を参照)。

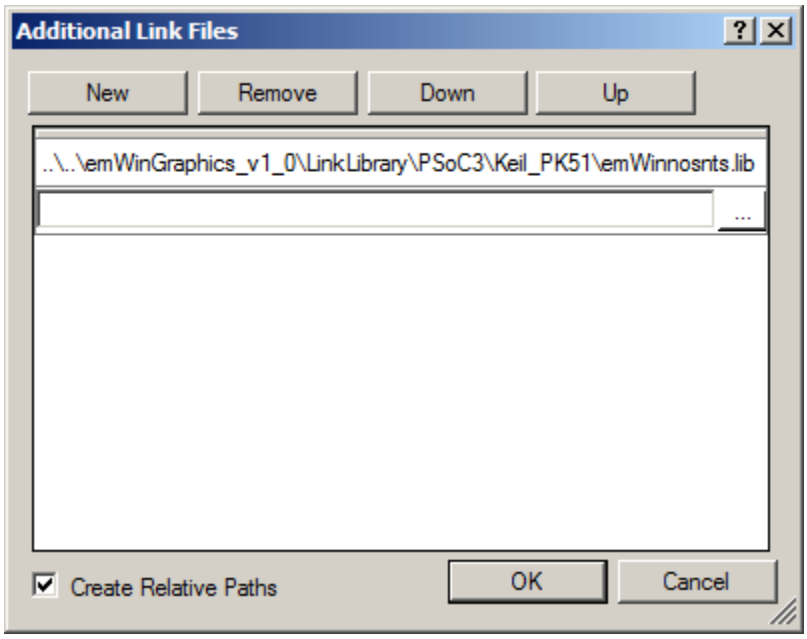
図 2. Additional Include Directories (追加のインクルード ディレクトリ) ダイアログ



3. リンク ライブラリ ファイルを追加します。emWin GUI は、Keil_PK51、GNU CC、KEIL MDK、または KEIL RVDS ツールチェーンで使用できるので、使用するツールチェーンに対応したライブラリを選択します。

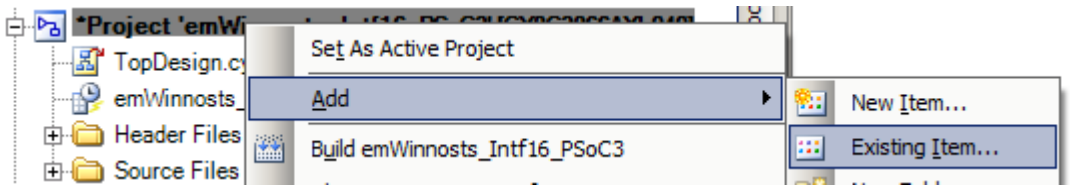
   メニューから **Project > Build Settings > Linker > General > Additional Link Files** を選択して、図 3 に示すライブラリ ファイルを選択します。

図 3. Additional Link Files (追加のリンク ファイル) ダイアログ

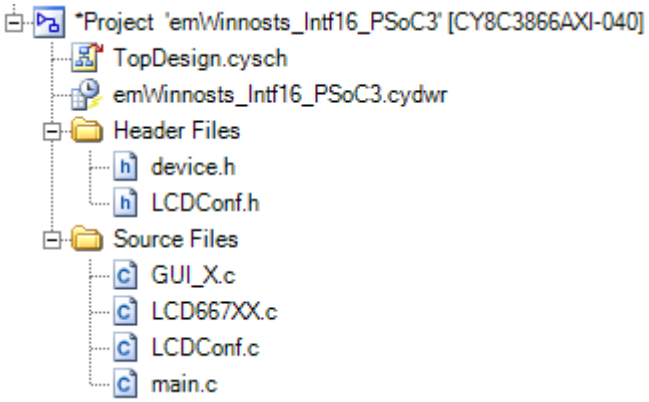


注 GCC ツールチェーンではライブラリをリンカーに追加するプロセスが異なります。この場合は、ライブラリを格納したディレクトリは「Additional Library Directories (追加のライブラリ ディレクトリ)」設定でリンカーに追加され、ライブラリは「Additional Libraries (追加のライブラリ)」設定で指定されます。この設定に使用されるライブラリ名からは、接頭辞「lib」および接尾辞「.a」がら削除される必要があります。

4. 選択した LCD コンポーネントのプロジェクトに必須のソース ファイルを追加します。Interface (インターフェイス) コンポーネントか Control (コントロール) コンポーネントのいずれかを選択します。その選択に応じて、 *Source\PSoC3\Graphics LCD Interface Parallel* ディレクトリか *Source\PSoC3\Graphics LCD Controller* ディレクトリのどちらかにあるファイルが有効になります。これらのファイルは、一般的にそれぞれのプロジェクト用に編集されるため、プロジェクトのルート フォルダにコピーしてから、プロジェクトに追加することをお勧めします。これらのファイルは、メニューの **Project (プロジェクト) > Add (追加) > Existing Item (既存アイテム)** を選択すると、プロジェクトに追加されます。インクルード ファイルがプロジェクト ディレクトリのルートに配置されていない場合は、インクルード ファイルのパスもビルド設定に追加する必要があります。



ここに示すプロジェクトには、 *Source\PSoC3\Graphics LCD Interface Parallel* ディレクトリから、*GUI_X_Touch.c* を除くすべてのファイルが追加されています。この例は OS もタッチスクリーンも使用していないため、タッチスクリーン関連ファイルは除外されています。

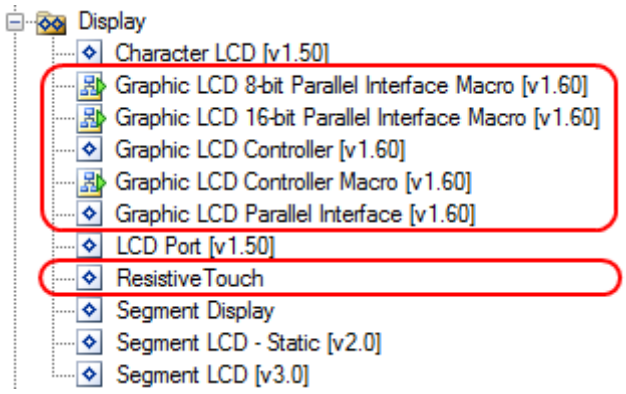


emWin とともに使用する PSoC Creator コンポーネント

emWinGraphics ライブラリはソフトウェア ライブラリです。これには LCD パネルを駆動するハードウェアは含まれていません。グラフィックス LCD コンポーネント (Graphics LCD Interface または Graphics LCD Controller) は、emWinGraphics ライブラリを使用して設計を構築するために必要です。いずれのコンポーネントも複数グラフィック パネルのサポートを提供しますが、それは別のタイプのパネルを駆動します。さらに、オプションのタッチスクリーン コンポーネントはタッチ ユーザ インターフェイスを実装するために emWin とともに使用できます。これらのコンポーネント間の通信は、完全にソフトウェアによって実行されます。

1 つの LCD グラフィックスデザインには Graphics LCD コンポーネントのインスタンスが 1 つだけ含まれ、オプションとして、タッチスクリーン コンポーネントのインスタンスが 1 つ含まれます。これらのコンポーネントはすべて、図 4 でハイライトされた Component Catalog の Display フォルダにあります。

図 4. Display フォルダ



Graphics LCD Interface (GraphicLCDIntf) コンポーネント

GraphicLCDIntf コンポーネントは、LCD パネルに一体化されたグラフィック LCD コントローラとドライバ デバイスを持ったパネルとのインターフェイスとして使用されます。また、このタイプのパネルには、パネルに一体化された LCD コントローラによって管理されるフレーム バッファが含まれています。このコンポーネントは、このコントローラへの読み取りおよび書き込み処理を実行します。

GraphicLCDIntf アプリケーションを作成するには、以下の手順が必要です:

1. Graphic LCD Parallel Interface Macro を回路図上に配置します。使用する LCD コントローラに応じて、8 ビットまたは 16 ビットのインターフェイスを選択します。
2. LCD コントローラへの書き込みおよび読み取り処理の要件を満たすように、GraphicLCDIntf コンポーネントを設定します。設定の詳細については、GraphicLCDIntf データシートを参照してください。
3. GraphicLCDIntf コンポーネントに固有のすべての emWin ファイルがあなたのプロジェクトに含まれていることを確認します。これらのファイルは、PSoC デバイスに応じて *Code\Source\PSoC3* か *Code\Source\PSoC5* ディレクトリの Graphics LCD Interface Parallel サブディレクトリにあります。
4. ご使用の LCD ディスプレイ ハードウェアに応じて、必要なソース ファイルを変更します。大半のソース ファイルは emWin のディストリビューションから変更せずに使用できますが、一部の設定ファイルはターゲット システムが正しく動作するように、変更する必要がありま

す。emWin 設定の詳細については、*emWin User Manual* の「Configuration (設定)」の章を参照してください。

Graphic LCD Interface アプリケーションに必要な emWin ファイル

GraphicLCDIntf コンポーネントを使用する PSoC 3 アプリケーションをビルドするには、*Code\Source\PSoC3\Graphics LCD Interface Parallel* ディレクトリにある次のファイルが必要です:

- *GUI_X.c*: このファイルを編集して、タイミング ルーチン (emWin ライブラリは時間についての情報を知る必要があります) およびデバッグ ルーチン (オプション) を与える必要があります。
- *GUI_X_Touch.c* (オプション): 抵抗膜方式タッチパネルインターフェイスを実装するためのハードウェア ルーチンを提供します。このファイルは編集する必要がありません。
- *LCD667XX.c*: ディスプレイ ドライバ。このファイルは編集する必要がありません。
- *LCDConf.h*: このファイルは、ディスプレイ ドライバのコンパイルに必要な全般的な設定オプションを与えるために編集する必要があります。使用できる設定オプションの詳細については、*emWin User Manual* の「Display drivers (ディスプレイ ドライバ)」の章を参照してください。
- *LCDConf.c*: このファイルは、使用する特定のディスプレイ コントローラの初期化ルーチンを与えるために、編集する必要があります。

GraphicLCDIntf コンポーネントを使用する PSoC 5 アプリケーションをビルドするには、*Code\Source\PSoC5\Graphics LCD Interface Parallel* フォルダにある次のファイルが必要です:

- *GUI_X.c*: このファイルを編集して、タイミング ルーチン (emWin ライブラリは時間についての情報を知る必要があります) およびデバッグ ルーチン (オプション) を与える必要があります。アプリケーションが OS を使用しない場合は、そのファイルをプロジェクトに含めます。OS を使用する場合は、このファイルの OS にあわせたバージョンを代わりに使用します。

- *GUI_X_embOS.c*: このファイルは、embOS RTOS に適合した GUI カーネル インターフェイス ルーチンを与えます。あなたのアプリケーションが embOS リアルタイム カーネルを使用する場合は、このファイルをあなたのプロジェクトに含めます。このファイルは、通常は編集する必要がありません。
- *GUI_X_uCOS.c*: このファイルは、uC/OS RTOS に適合した GUI カーネル インターフェイス ルーチンを与えます。あなたのアプリケーションが uC/OS リアルタイム カーネルを使用する場合は、このファイルをあなたのプロジェクトに含めます。このファイルは、通常は編集する必要がありません。
- *GUI_X_Touch.c* (オプション): 抵抗膜式タッチパネル インターフェイスを実装するためのハードウェア ルーチンを提供します。このファイルは編集する必要がありません。
- *GUIConf.c*: emWin オペレーションにメモリを割り当てます。このファイルは、特定のチップやアプリケーション用に調整できます。デフォルト設定は、emWin が使用する 8,192 バイトの SRAM です。
- *GUIDRV_CompactColor_16.c*: ディスプレイ ドライバ。このファイルは編集する必要がありません。
- *LCDConf.h*: Compact Color Driver が使用されていることを示します。このファイルは編集する必要がありません。
- *LCDConf_CompactColor_16.h*: このファイルは、ディスプレイ ドライバのコンパイルに必要な全般的な設定オプションを与えるために編集する必要があります。使用できる設定オプションの詳細については、*emWin User Manual* の「Display drivers (ディスプレイ ドライバ)」の章を参照してください。
- *LCDConf.c*: このファイルは、使用する特定のディスプレイ コントローラの初期化ルーチンを与えるために編集する必要があります。

Graphics LCD Controller (GraphicLCDCtrl) コンポーネント

GraphicLCDCtrl コンポーネント は、LCD ドライバがあるが、LCD コントローラがない LCD パネルへのインターフェースです。この種のパネルには、フレームバッファがありません。フレ

ームバッファを外部で提供する必要があります。このコンポーネントは、制御信号を直接駆動し、外部 SRAM 内でフレームバッファを管理します。

GraphicLCDCtrl アプリケーションを作成するには、以下の手順が必要です

1. Graphic LCD Controller Macro を回路図上に配置します。
2. ご使用の LCD パネルのスクリーン リフレッシュ要件とデータトランザクションタイミング パラメータを満たすように、GraphicLCDCtrl コンポーネントを設定します。設定の詳細については、GraphicLCDCtrl データシートを参照してください。
3. GraphicLCDCtrl コンポーネントに固有のすべての emWin ファイルがプロジェクトに含まれていることを確認します。これらのファイルは、使用予定の PSoC デバイスに応じて *Code\Source\PSoC3* か *Code\Source\PSoC5* ディレクトリの Graphics LCD Interface Controller サブディレクトリにあります。
4. ご使用の LCD ディスプレイ ハードウェアに応じて、必要なソース ファイルを変更します。大半のソース ファイルは emWin の分配を変更せずに使用できますが、一部の設定ファイルはターゲット システムが正しく動作するように、編集する必要があります。emWin の設定の詳細については、*emWin User Manual* の「Configuration (設定)」の章を参照してください。

Graphic LCD Controller アプリケーションに必要な emWin ファイル

GraphicLCDCtrl コンポーネントを使用して PSoC 3 アプリケーションをビルドするには、*Code\Source\PSoC3\Graphics LCD Controller* ディレクトリにある次のファイルが必要です:

- *GUI_X.c*: このファイルを編集して、タイミング ルーチン (emWin ライブラリは時間についての情報を知る必要があります) およびデバッグ ルーチン (オプション) を与える必要があります。
- *GUI_X_Touch.c* (オプション): 抵抗膜式タッチパネル インターフェイスを実装するためのハードウェア ルーチンを提供します。このファイルは編集する必要がありません。
- *GUIDRV_Control.c*: ディスプレイ ドライバ。このファイルは編集する必要がありません。
- *LCDConf.h*: このファイルは、ディスプレイ ドライバのコンパイルに必要な全般的な設定オプションを与えるために編集する必要があります。使用できる設定オプションの詳細につい

ては、*emWin User Manual* の「Display drivers (ディスプレイ ドライバ)」の章を参照してください。

- *LCDConf.c*: このファイルは、使用する特定の LCD パネルの初期化ルーチンを与えるために編集する必要があります。

GraphicLCDCtrl コンポーネントを使用して PSoC 5 アプリケーションをビルドするには、*Code\Source\PSoC5\Graphics LCD Controller* ディレクトリにある次のファイルが必要です:

- *GUI_X.c*: このファイルを編集して、タイミング ルーチン (emWin ライブラリは時間についての情報を知る必要があります) およびデバッグ ルーチン (オプション) を与える必要があります。アプリケーションが OS を使用しない場合は、このファイルをプロジェクトに含めます。OS を使用する場合は、OS にあわせたこのファイルのバージョンを代わりに使用します。
- *GUI_X_embOS.c*: このファイルは、embOS RTOS に適合した GUI カーネル インターフェイス ルーチンを与えます。あなたのアプリケーションが embOS リアルタイム カーネルを使用する場合は、このファイルをあなたのプロジェクトに含めます。このファイルは、通常は編集する必要がありません。
- *GUI_X_uCOS.c*: このファイルは、uC/OS RTOS に適合した GUI カーネル インターフェイス ルーチンを与えます。あなたのアプリケーションが uC/OS リアルタイム カーネルを使用する場合は、このファイルをあなたのプロジェクトに含めます。このファイルは、通常は編集する必要がありません。
- *GUI_X_Touch.c* (オプション): 抵抗膜式タッチパネル インターフェイスを実装するためのハードウェア ルーチンを提供します。このファイルは編集する必要がありません。
- *GUIConf.c*: emWin オペレーションにメモリを割り当てます。このファイルは、特定のチップやアプリケーション用に調整できます。デフォルト設定は、emWin が使用する 8,192 バイトの SRAM です。
- *GUIDRV_Control.c*: ディスプレイ ドライバ。このファイルは編集する必要がありません。
- *LCDConf.h*: 使用されるドライバを示し、表示方向や色深度などの一部の全般的設定オプションが含まれます。このファイルは必要に応じて編集できます。

- *LCDConf.c*: このファイルは、使用する特定の LCD パネルの初期化ルーチンを与えるために編集する必要があります。

アナログ タッチスクリーンのセットアップ

タッチスクリーンを接続する最も一般的な方法は、4 ピン アナログ インターフェースです。このためのドライバは emWin によって供給されます。emWin タッチスクリーン ドライブは、アナログ入力 (A/D コンバータで取得)、デバウンス、およびタッチスクリーンのキャリブレーションを管理します。

タッチスクリーン ドライバは、関数 GUI_TOUCH_Exec() を使用して、タッチ パネルを継続的にモニターし、更新します。この関数は、アクションが実行されたり、何かが変更されたことを認識すると、適切な汎用タッチスクリーン API ルーチンを呼び出します。

以下の手順を使用して、タッチスクリーン機能をあなたのプロジェクトに追加します:

1. タッチスクリーン サポートを持った emWin ライブラリを使用するためにプロジェクト設定が行われていることを確認します。emWin ライブラリを選択する方法についての詳細は、本書の emWinGraphics Library を PSoC Creator に統合する のセクションを参照してください。
2. *GUI_X_Touch.c* ファイルをあなたのプロジェクトに追加します。
3. Resistive Touch コンポーネントを回路図上に配置します。このコンポーネントは Component Catalog の Display フォルダにあります。
4. GUI_TOUCH_Exec() への定期的な呼び出しを実装します。あなたのアプリケーションはこれを約 100 回/秒で呼び出す必要があります。リアルタイム オペレーティング システムを使用している場合、この関数が確実に呼び出されれるようにする最も簡単な方法は、独立したタスクを作成することです。マルチタスキング システムを使用しない場合は、割り込みサービス ルーチンを使用できます。
5. 使用するタッチパネルの、A/D コンバータから返される最小値と最大値を調べ、これらの値で関数 GUI_TOUCH_Calibrate() のパラメータを変更します。emWin は、測定結果をピクセル単位のタッチ位置に変換するためにこれらの値を必要とします。タッチスクリーン設定の詳細については、*emWin User Guide* の 19.4.2 章「The analog touch screen driver (アナロ

グ タッチスクリーン ドライバ)」を参照してください。ユーザ ガイドで引用されているファイル *TOUCH_Sample.c* および *TOUCH_Calibrate.c* は、*Resources\Examples* ディレクトリにあります。

EmWin サンプル ファイル

emWinGraphics コンポーネントは、PSoC 3 および PSoC 5 に使用できる多くの emWin サンプル ファイルとともに出荷されています。これらは、*Resources\Samples* ディレクトリの PSoC 3 または PSoC 5 サブディレクトリにあります。emWin に慣れていない場合は、独自のアプリケーションを作成する前に、サンプル プログラムをコンパイル、リンク、テストすることをお勧めします。

サンプル ファイルを実行するには:

1. デモ サンプル ファイルのいずれかをあなたのプロジェクトに追加します。
2. メイン ソース ファイル (*main.c*) を編集して *GUI.h* をインクルードします。割り込みを有効にして、以下のコードに示されているように MainTask() を呼び出します。すべての emWin の例は、エントリ ポイントとして MainTask() を使用します。

```
#include <device.h>
#include "GUI.h"

extern void MainTask(void);

void main()
{
    CyGlobalIntEnable;
    MainTask();

    for(;;)
    {
        /* あなたのアプリケーションのコードをここに入れる。*/
    }
}
```

3. プロジェクトをビルドして実行する。

使用できるライブラリ

emWin GUI は Keil、PK51、GNU CC、KEIL MDK、または KEIL RVDS ツールチェーンとともに使用できます。Keil_PK51 を除くすべてのツールチェーンで、ライブラリは OS (os) サポートとタッチスクリーン (ts) サポートの 2 つのオプションに使用できます。OS サポートは、Graphics ライブラリの初期化時に必要なフックと、OS によるライブラリ内リソースのロックとロック解除用フックを提供します。特定のライブラリ名によってサポートされるオプションを以下に示します:

オプション	OS サポート	TS サポート
nosnts	NO	NO
nosts	NO	YES
osnts	YES	NO
osts	YES	YES

Keil_PK51 ツールチェーン用の emWin ライブラリ

このライブラリは *LinkLibrary\PSoC3\Keil_PK51* ディレクトリにあり、次のように命名されます:

emWin<options>.lib

使用できるオプションは nosnts および nosts です。

例:

libemWinnosts.a は、OS サポートがなく、タッチスクリーン サポートがあるライブラリです。

GCC ツールチェーン用の emWin ライブラリ

このライブラリは *LinkLibrary\PSoC5\GCC* ディレクトリにあり、次のように命名されます:

libemWin<options>.a

使用できるオプションは nosnts、nosts、osnts、osts です。

例:

libemWinosnts.a は、OS サポートがあり、タッチスクリーン サポートがないライブラリです。

GCC ツールチェーンを使用する場合は、ライブラリ名の接頭辞「lib」とファイル拡張子「.a」を **Linker** (リンカー) オプションの **Additional Libraries** (追加ライブラリ) フィールドで削除する必要があることに注意してください。

Keil MDK ツールチェーン用の emWin ライブラリ

このライブラリは *LinkLibrary\PSoC5\ARM_MDK* ディレクトリにあり、次のように命名されます:

emWin<options>.lib

使用できるオプションは nosnts、nosts、osnts、osts です。

例:

emWinosts.lib は OS とタッチスクリーン サポートを持ったライブラリです。

ARM RVDS ツールチェーン用の emWin ライブラリ

このライブラリは *LinkLibrary\PSoC5\ARM_RVDS* ディレクトリにあり、次のように命名されます:

emWin<options>.lib

使用できるオプションは nosnts、nosts、osnts、osts です。

例:

emWinnosnts.lib は OS とタッチスクリーン サポートを持たないライブラリです。

配置

実装はすべてソフトウェアです。

性能とリソース利用

実際の性能とリソース使用は多くの要素 (CPU、コンパイラ、メモリ モデル、最適化、設定、LCD コントローラへのインターフェースなど) によって異なります。このセクションには、一般的な設定に対するイメージ描画動作の性能値とメモリ リソース使用についての情報が記載されています。

性能

製品	CPUの速度 (MHz)	Graphic LCD コンポーネント	bpp	フィリング[1]	小さなフォント[2]	大きなフォント[2]	ビットマップ 1 bpp	ビットマップ 8 bpp	ビットマップ 16 bpp
PSoC 3	24	グラフィックス LCD インターフェース (16 bit)	16	112K	24K	30K	36K	14K	適用外
PSoC 3	60	グラフィックス LCD インターフェース (16 bit)	16	279K	56K	70K	82K	33K	適用外
PSoC 3	24	グラフィックス LCD インターフェース (8 bit)	16	63K	20K	24K	28K	13K	適用外
PSoC 3	60	グラフィックス LCD インターフェース (8 bit)	16	156K	46K	56K	65K	30K	適用外

[1] 表に示された性能値とご使用のディスプレイの解像度に基づいて、以下のように性能値を 1 秒あたりのフレーム数 (fps) で計算できます:

fps = 1 秒あたりのピクセル数 / (LCD_XSIZE × LCD_YSIZE)

[2] これらの値に基づいて、1 秒あたりのおおよその描画文字数を以下のように計算できます :

1 秒あたりの文字数 = 1 秒あたりのピクセル数 / (文字幅 × 文字高さ)

文字サイズの詳細については、*emWin User Manual* の「Fonts (フォント)」のセクションを参照してください。

製品	CPUの速度 (MHz)	Graphic LCD コンポーネント	bpp	フィリング[1]	小さなフォント[2]	大きなフォント[2]	ビットマップ 1 bpp	ビットマップ 8 bpp	ビットマップ 16 bpp
PSoC 3	24	グラフィック LCD コントローラ	16	34K	21K	23K	26K	12K	適用外
PSoC 3	60	グラフィック LCD コントローラ	16	75K	48K	54K	61K	29K	適用外
PSoC 5	24	グラフィックス LCD インターフェース (16 bit)	16	662K	156K	177K	195K	150K	589K
PSoC 5	60	グラフィックス LCD インターフェース (16 bit)	16	1.64M	315K	357K	400K	306K	1.40M
PSoC 5	24	グラフィックス LCD インターフェース (8 bit)	16	300K	124K	136K	149K	117K	178K
PSoC 5	60	グラフィックス LCD インターフェース (8 bit)	16	570K	231K	252K	277K	219K	335K
PSoC 5	24	グラフィック LCD コントローラ	16	147K	131K	130K	135K	114K	152K
PSoC 5	60	グラフィック LCD コントローラ	16	224K	200K	192K	207K	184K	229K

メモリ必要条件

emWin ライブラリのメモリ使用は、主として使用されるアプリケーションと機能に応じて大きく異なります。次の表は、メモリ必要条件についての基本的な考え方を提示するため、一部のサンプル アプリケーションの全体サイズ (コード、フォント、ブート、イメージなどを含むすべて) を提供します。より厳密な計算を行うには、emWin ライブラリの SEGGER ドキュメンテーションを参照してください。

サンプル	Keil 8051		GCC-4.4.1		説明
	フラッシュ	SRAM	フラッシュ	SRAM	
Hello World アプリケーション	19.4 KB	3.6 KB	24.5 KB	2.9 KB	このサンプル プロジェクトは、テキストを表示するための基本的な関数のみを使用します。
2D Graphics アプリケーション	33 KB	3.7 KB	34 KB	2.9 KB	このサンプルは、様々なフォントを使用してテキストを表示し、2D Graphics Library の一部の描画動作を実行します。
ウィンドウ アプリケーション	該当せず	該当せず	84 KB	11 KB	このプロジェクトは、ウィンドウ マネージャと様々なウィジェットを使用する「一般的な」アプリケーションです。

アプリケーション プログラミング インタフェース

emWinGraphics コンポーネントには、標準 emWin ライブラリに存在しない API 関数は含まれていません。 使用できる emWin ルーチンの詳細な説明については、SEGGER emWin のドキュメンテーションを参照してください。

機能の説明

詳しい機能については、SEGGER emWin ドキュメンテーションに記載されています。このセクションでは、PSoC 3 および PSoC 5 についての SEGGER 機能の主なカテゴリについて説明しています。

SEGGER emWin Library 機能	PSoC 3	PSoC 5
2D グラフィックス ライブラリ	フルサポート	フルサポート
ビットマップ ファイルの表示	コンパイルしたビットマップ形式以外の形式のサポートなし。 16 ビット ビットマップのサポートなし。8 ビット以下のビットマップは完全にサポートされています。	コンパイルした全てのビットマップ形式のサポート その他の形式 (JPEG、GIF、PNG) もサポート可能ですが、必要となるRAM容量の面から実用的ではありません
フォント	プロポーショナル フォントのみをサポート	すべてのフォント タイプをサポート
ビットマップ コンバータ	バイナリ形式で出荷されるコンパイル済みウィンドウ アプリケーション。	
カラー サポート	フルサポート	フルサポート
メモリ デバイス	サポートなし	フルサポート
マルチタスク (RTOS)	サポートなし	フルサポート
ウィンドウ マネージャ	サポートなし	フルサポート
ウィンドウ オブジェクト (ウィジェット)	サポートなし	フルサポート
仮想スクリーン/仮想ページ	フルサポート	フルサポート
マルチレイヤ/マルチディスプレイ	サポートなし	サポートなし
ポインタ入力デバイス	タッチスクリーンのみをフルサポート	タッチスクリーンのみをフルサポート
スプライトとカーソル	サポートなし	フルサポート
アンチエイリアシング	サポートなし	フルサポート
外国語	サポートなし	サポートなし
VNC	サポートなし	サポートなし
ドライバ	コンパクト カラー ドライバとサイプレス カスタム ドライバ	

コンポーネント変更履歴

バージョン 1.0 は emWinGraphics ライブラリの最初のリリースです。

Copyright © 2005-2012 Cypress Semiconductor Corporation　本文書に記載される情報は、予告なく変更される場合があります。Cypress Semiconductor Corporationは、サイプレス製品に組み込まれた回路以外のいかなる回路を使用することに対しても一切の責任を負いません。特許又はその他の権限下で、ライセンスを譲渡又は暗示することもありません。サイプレス製品は、サイプレスとの書面による合意に基づくものでない限り、医療、生命維持、救命、重要な管理、又は安全の用途のために仕様することを保証するものではなく、また使用することを意図したものでもありません。さらにサイプレスは、誤動作や故障によって使用者に重大な傷害をもたらすことを合理的に予想される、生命維持システムの重要なコンポーネンツとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。

PSoC Designer™及びProgrammable System-on-Chip™は、Cypress Semiconductor Corp.の商標、PSoC®は同社の登録商標です。本文書で言及するその他全ての商標又は登録商標は各社の所有物です。

全てのソースコード(ソフトウェア及び/又はファームウェア)はCypress Semiconductor Corporation (以下「サイプレス」)が所有し、全世界(米国及びその他の国)の特許権保護、米国の著作権法並びに国際協定の条項により保護され、かつそれらに従います。サイプレスが本書面によるライセンシーに付与するライセンスは、個人的、非独占的かつ譲渡不能のライセンスであって、適用される契約で指定されたサイプレスの集積回路と併用されるライセンシーの製品のみをサポートするカスタムソフトウェア及び/又はカスタムファームウェアを作成する目的に限って、サイプレスのソースコードの派生著作物を複製、使用、変更、そして作成するためのライセンス、並びにサイプレスのソースコード及び派生著作物をコンパイルするためのライセンスです。上記で指定された場合を除き、サイプレスの書面による明示的な許可なくして本ソースコードを複製、変更、変換、コンパイル、又は表示することは全て禁止されます。

免責条項：サイプレスは、明示的又は黙示的を問わず、本資料に関するいかなる種類の保証も行いません。これには、商品性又は特定目的への適合性の黙示的な保証が含まれますが、これに限定されません。サイプレスは、本文書に記載される資料に対して今後予告なく変更を加える権利を留保します。サイプレスは、本文書に記載されるいかなる製品又は回路を適用又は使用したことによって生ずるいかなる責任も負いません。サイプレスは、誤動作や故障によって使用者に重大な傷害をもたらすことが合理的に予想される生命維持システムの重要なコンポーネンツとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。

ソフトウェアの使用は、適用されるサイプレスソフトウェアライセンス契約によって制限され、かつ制約される場合があります。