RX ファミリ

emWin v6.52 モジュール Firmware Integration Technology

要旨

本アプリケーションノートは、Firmware Integration Technology (FIT)を使用した emWin v6.52 モジュールについて説明します。以降、本モジュールを emWin FIT モジュールと称します。

emWin FIT モジュールは SEGGER 社の emWin（https://www.segger.com/products/user-interface/emwin/add-ons/emwin-support-renesas-rx-mcu/）を FIT モジュール化したものです。

本 emWin FIT モジュールを RX ファミリ（FIT モジュールがサポートしている製品）でご使用いただく場合、基本的にはライセンス不要で量産いただくことが可能です。ライセンスの詳細は「1.1 emWin とは」を参照ください。

「emWin」および GUI デザインツール「AppWizard」の詳細に関しては、SEGGER 社（https://www.segger.com/）または国内代理店のエンビテック社（https://www.embitek.co.jp/）にお問い合わせください。

本モジュールは、ディスプレイ対応開発支援ツール「QE for Display[RX]」(https://www.renesas.com/jp/ja/software-tool/qe-display-development-assistance-tool-display-applications)と連携しています。emWin FIT モジュールをご使用の際は「QE for Display[RX]」も併わせてご使用いただくことを推奨します。

対象デバイス

・RX ファミリ

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様に合わせて変更し、十分評価してください。

対象コンパイラ

・Renesas Electronics C/C++ Compiler Package for RX Family

・GCC for Renesas RX

・IAR C/C++ Compiler for Renesas RX

各コンパイラの動作確認内容は 8.1 動作確認環境を参照してください。

関連ドキュメント

- Firmware Integration Technology ユーザーズマニュアル(R01AN1833)
- ボードサポートパッケージモジュール Firmware Integration Technology (R01AN1685)

目次

1. 概要

1.1 emWin とは

emWin は Segger 社が提供する GUI ライブラリです。API レベルのプログラミングでも容易に GUI をデザインできる他、別途提供されている AppWizard などの各種ツールと組み合わせて使用することでより効率的に開発をすることができます。

emWin をお客様自身でご使用いただく場合、通常 Segger 社とのライセンス契約が必要となりますが、emWin FIT モジュールを RX ファミリ（FIT モジュールがサポートしている製品）でご使用いただく場合、そのままの構成で使用されることを条件にライセンス不要で量産化いただくことが可能です。

ただし、emWin FIT モジュールに含まれていないディスプレイドライバや機能、ライブラリのソースコードなどが必要になる場合、別途 Segger 社とのライセンス契約が必要となります。

ライセンス契約の詳細は Segger 社または国内代理店のエンビテック社（https://www.embitek.co.jp/）にお問い合わせください。

1.2 emWin FIT モジュールの概要

emWin FIT モジュールは、SEGGER 社の emWin（v6.52）を FIT 化することで、スマート・コンフィグレータにより簡単にユーザプログラムへ emWin を組み込めるようにしたものです。v6.52 以降のバージョンアップにも対応して行く予定です。

emWin の詳細は以下のドキュメントを参照してください。

・emWin Graphic Library with Graphical User Interface User Guide & Reference Manual

（https://www.segger.com/downloads/emwin/UM03001）

emWin FIT モジュールは、以下のインタフェースに対応しています。

＜LCD＞

・RGB（パラレルインタフェース）

RX65N グループ、RX72N グループなどの GLCDC を搭載した RX 製品で使用できます。

emWin において「GUIDRV_LIN」ドライバに対応した制御となります。

・SPI（シリアルインタフェース）

RSPI、SCI（簡易 SPI モード）を搭載した RX 製品で使用できます。

emWin において「GUIDRV_FlexColor」ドライバに対応した制御となります。

注.「GUIDRV_LIN」および「GUIDRV_FlexColor」は emWin で定義されたディスプレイドライバ群の名称です。

＜タッチパネル＞

・I^2C

・SPI

1.3　制限事項

emWin FIT モジュールには、以下の制限事項があります。

・DRW2D FIT モジュールの使用を推奨（DRW2D は GLCDC による RGB（パラレルインタフェース）でのみ使用できます）

・OS は FreeRTOS、OS レスのみ対応

・SEGGER 社製の emFILE および embOS には非対応

・「GUIDRV_LIN」ドライバおよび「GUIDRV_FlexColor」ドライバのみサポート

・「8.1 動作確認環境」に記載した LCD 以外は動作未確認（動作未確認の LCD をご使用される場合、「6.4 動作確認環境以外での使用」を参照してください）

・GCC コンパイラを使用される場合、倍精度浮動小数点処理命令に関するビルドオプション(-mdfpu と-m64bit-doubles)は使用禁止

・IAR コンパイラを使用される場合、C/C++ランタイムライブラリには「通常の DLIB」を選択してください。

・リトルエンディアンにのみ対応

1.4　製品構成

本製品は、以下の表 1-2 のファイルが含まれます。

表 1.1　製品構成

ファイル/ディレクトリ(**太字**)名	内容
r01an8024jj0100-rx-emwin.pdf	emWin FIT モジュール アプリケーションノート(日本語)
r01an8024ej0100-rx-emwin.pdf	emWin FIT モジュール アプリケーションノート(英語)
FITModules	FIT モジュールフォルダ
r_emwin_rx_v6.52_100.mdf	emWin FIT モジュール スマート・コンフィグレータ用コンフィグレーション設定ファイル
r_emwin_rx_v6.52_100.xml	emWin FIT モジュール e² studio FIT プラグイン用 XML ファイル
r_emwin_rx_v6.52_100.zip	emWin FIT モジュール （以下は展開時のフォルダ内容物）
r_config	emWin FIT モジュール用のコンフィグ H ファイル格納フォルダ
r_emwin_rx_config.h	emWin FIT モジュール用のコンフィグ H ファイル
r_emwin_rx	emWin FIT モジュールのソースコードとドキュメントおよび関連ツール格納用フォルダ
readme.txt	emWin FIT モジュールの概要とファイル構成についての説明
r_emwin_rx_if.h	emWin FIT モジュール用の宣言 H ファイル
Doc	emWin FIT モジュールのドキュメント格納フォルダ
emWin_doc	SEGGER 社提供の、 emWin ライブラリに関するドキュメント格納用フォルダ
en	emWin FIT モジュール アプリケーションノート(英語)格納用フォルダ
r01an8024ej0100-rx-emwin.pdf	emWin FIT モジュール アプリケーションノート(英語)
ja	emWin FIT モジュール アプリケーションノート(日本語)格納用フォルダ
r01an8024jj0100-rx-emwin.pdf	emWin FIT モジュール アプリケーションノート(日本語)
Training	SEGGER 社提供の、emWin ライブラリを使用するサンプルプログラム (詳細は、本フォルダに格納されている emWin_Training.pdf を参照してください。)
lib	emWin ライブラリ、およびこれに対するコンフィグレーションとして用意しているインタフェース部のソースコードとヘッダファイル格納用フォルダ
Config	emWin ライブラリに対するインタフェース部のソースコードとヘッダファイル格納用フォルダ
APPW_X_NoFS.c	AppWizard を使用するための関数を含む C ファイル (※ファイルシステムを使用しない場合の関数のみサポートしています)
GUI_X_Ex.c	RTOS の使用時、および非使用時に使用する関数を含む C ファイル
GUIConf.c	emWin ライブラリが使用するワークメモリの確保と初期化に使用する関数を含む C ファイル

GUIConf.h	emWin ライブラリのコンフィグレーションを示す H ファイル
LCDConf_glcdc_if.c	emWin ライブラリのディスプレイドライバの使用と初期化、および GLCDC や DRW2D による制御処理を含む C ファイル
LCDConf_sci_spi_if.c	emWin ライブラリのディスプレイドライバの使用と初期化、および SCI の簡易 SPI モードによる制御処理を含む C ファイル
LCDConf_rspi_if.c	emWin ライブラリのディスプレイドライバの使用と初期化、および RSPI による制御処理を含む C ファイル
LCDConf_user_if.c	ユーザ定義の処理を実装する C ファイル 上記 GLCDC や SPI の制御では動作しない場合、本ファイルにユーザ定義の処理を実装します。
LCDConf.h	ディスプレイドライバの使用に関する H ファイル
PIDConf.c	タッチ機能の使用と初期化を担う関数を含む C ファイル
PIDConf.h	タッチ機能の使用に関する H ファイル
GUI	emWin ライブラリのライブラリファイルとヘッダファイル格納用フォルダ
src	emWin ライブラリに対するインタフェース部以外のソースコードとヘッダファイル格納用フォルダ
r_emwin_rx_if.c	emWin FIT モジュール固有の関数を含む C ファイル
r_emwin_rx_pid_iic_if.c	タッチパネルとの I²C インタフェース機能を担う C ファイル
r_emwin_rx_pid_spi_if.c	タッチパネルとの SPI インタフェース機能を担う C ファイル
r_emwin_rx_pid_user_if.c	ユーザ定義の処理を実装する C ファイル 上記 I²C や SPI の制御では動作しない場合、本ファイルにユーザ定義の処理を実装します。
r_emwin_rx_private.h	emWin FIT モジュール内部で使用する H ファイル
r_emwin_rx_lcd_driver_ili9341.h	LCD ドライバ ILI9341 の定義に関する H ファイル
r_emwin_rx_lcd_driver_st7715.h	LCD ドライバ ST7715 の定義に関する H ファイル
tool	AppWizard インストーラを始めとする emWin ライブラリを使用するためのツール類格納フォルダ (詳細は、doc/Training フォルダに格納されている emWin_Training.pdf を参照してください。)

1.5　API の概要

emWin FIT モジュールに含まれる API 関数を以下に示します。表 1.2 に emWin が内部で呼び出す関数一覧を、表 1.3 にアプリケーションから呼び出される API 関数一覧を示します。詳細は 3 章 emWin から呼び出される関数、4 章アプリケーションから呼び出される API 関数を参照してください。

表 1.2 emWin が内部で呼び出す関数一覧

関数	関数説明
GUI_X_Config	emWin のメモリ管理システムで使用するメモリブロックの登録に使用します。
LCD_X_Config	LCD とデバイスドライバの初期化に使用します。
LCD_X_DisplayDriver	ディスプレイドライバのコールバック関数として使用します。
GUI_X_Init	必要なハードウェアを初期化します。
GUI_X_Delay	指定の時間を待ちます。
GUI_X_ExecIdle	Window Manager から、GUI が最新の状態で処理すべき内容がないときに呼び出されます。
GUI_X_GetTime	現在のシステム時間がミリ秒単位の整数型で得られます。
GUI_X_ErrorOut	致命的なエラーが発生した際に、エラー文字列を入力として emWin から呼び出されます。
GUI_X_Warn	警告が発生した際に、警告文字列を入力として emWin から呼び出されます。
GUI_X_Log	メッセージが発生した際に、メッセージ文字列を入力として emWin から呼び出されます。
GUI_X_InitOS	マルチタスク環境で使用する場合に、セマフォまたは mutex を生成します。
GUI_X_Unlock	マルチタスク環境で使用する場合に、GUI をアンロックします。
GUI_X_Lock	マルチタスク環境で使用する場合に、GUI をロックします。
GUI_X_GetTaskId	マルチタスク環境で使用する場合に、タスク ID を取得します。
GUI_X_WaitEvent	マルチタスク環境で使用する場合に、イベント待ちを実施します。
GUI_X_SignalEvent	マルチタスク環境で使用する場合に、イベント通知を実施します。
GUI_X_WaitEventTimed	マルチタスク環境で使用する場合に、指定した期間のイベント待ちを実施します。
PID_X_SetLayerIndex	レイヤ番号を設定します。
PID_X_Init	Pointer Input Device を初期化します。
GUI_TOUCH_X_ActiveX	Touch IC の X 軸の電圧測定を有効にします。
GUI_TOUCH_X_ActiveY	Touch IC の Y 軸の電圧測定を有効にします。
GUI_TOUCH_X_MeasureX	Touch IC から得られる X 軸の電圧測定結果を返します。
GUI_TOUCH_Y_MeasureY	Touch IC から得られる Y 軸の電圧測定結果を返します。
APPW_X_FS_Init	AppWizard のファイルシステムアクセスを初期化します。

表 1.3 アプリケーションから呼び出される API 関数一覧

関数	関数説明
R_EMWIN_GetBufferAddr	フレームバッファのアドレスを取得します。
R_EMWIN_GetD2	Dave2D 機能のハンドルを取得します。
R_EMWIN_EnableDave2D	Dave2D 機能を動作許可状態にします。
R_EMWIN_DisableDave2D	Dave2D 機能を動作禁止状態にします。
R_EMWIN_GetDaveActive	Dave2D 機能の動作状態を取得します。
R_EMWIN_GetVersion	emWin のバージョンを取得します。
_VSYNC_ISR()	Vsync 割り込み処理を行います。（GLCDC のコールバック関数を想定）

1.6　ソフトウェア構成

emWin FIT モジュールを使用したアプリケーションは、図 1.1 のようなソフトウェア構成になります。

アプリケーションからは emWin FIT モジュールを使用します。

RGB（パラレルインタフェース）の LCD を使用する場合、emWin FIT モジュールから DRW2D FIT モジュールを使用して図形を生成し、GLCDC FIT モジュールを使用して LCD への表示を行います。

SPI（シリアルインタフェース）の LCD を使用する場合、RSPI または SCI FIT モジュールを使用して LCD への表示を行います。

タッチパネル情報は簡易 I²C (SCI-IIC) FIT モジュールまたは RSPI FIT モジュール、SCI FIT モジュールを使用して制御します。

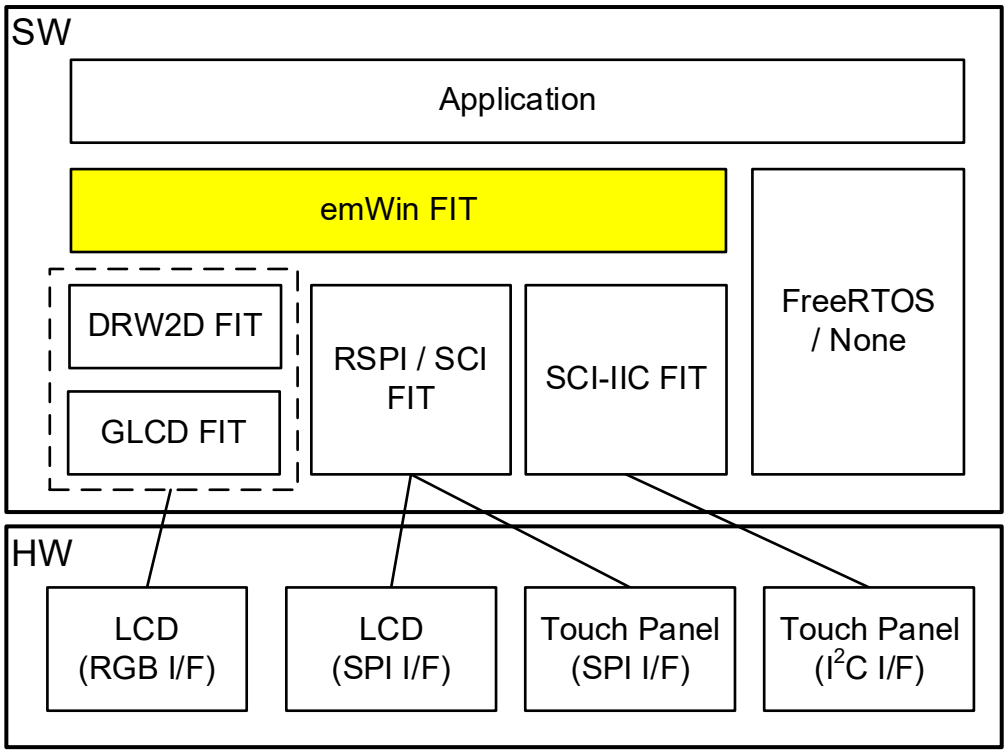


図 1.1 ソフトウェア構成

1.7　emWin のインタフェース構成

emWin FIT モジュールおよび各周辺モジュール、各種ツールとのインタフェース構成を図 1.2 と図 1.3 に示します。

○RGB（パラレルインタフェース）の場合

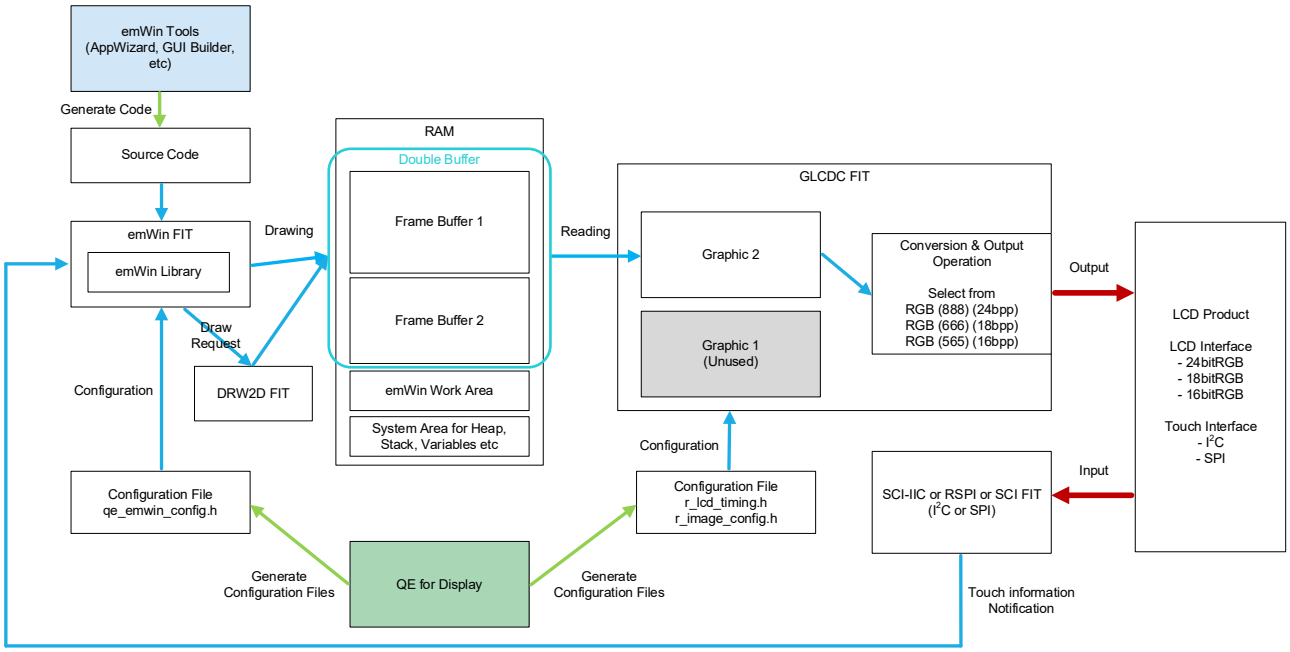


図 1.2 RGB（パラレルインタフェース）構成

○SPI（シリアルインタフェース）の場合

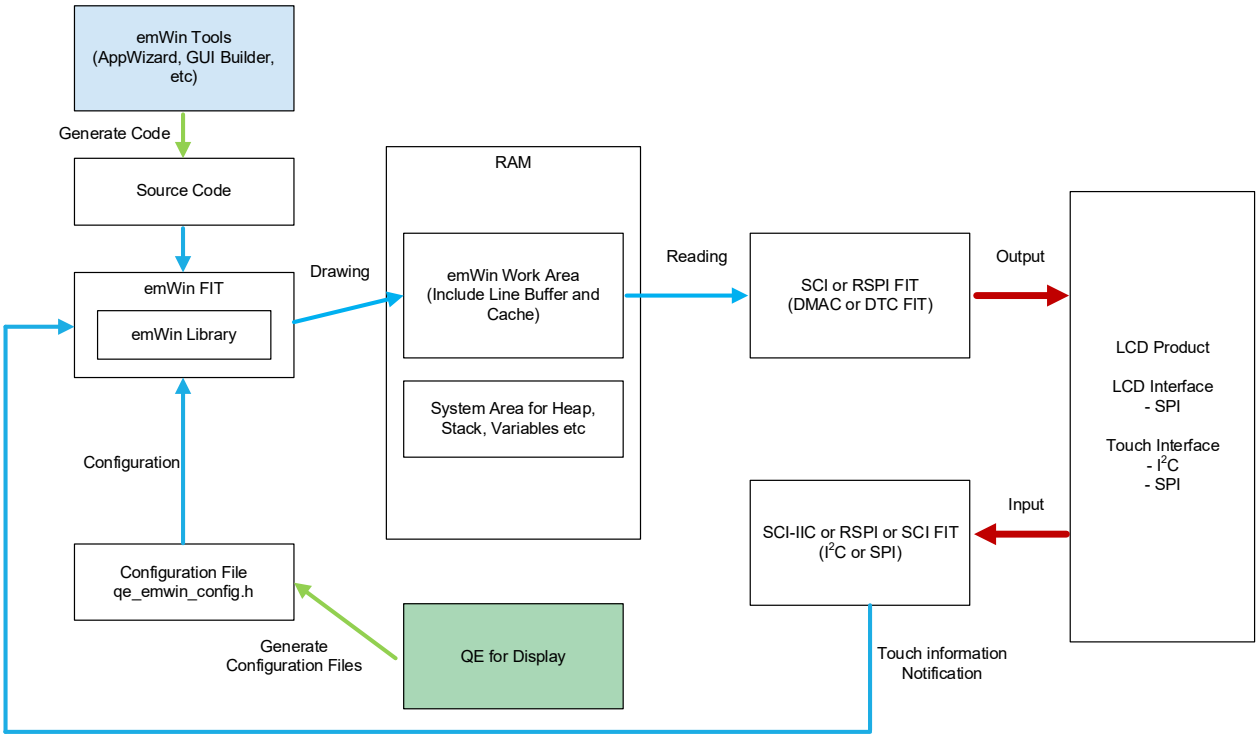


図 1.3 SPI（シリアルインタフェース）構成

1.8　色深度

LCD のインタフェースによって設定可能な色深度が異なります。これは、emWin が用意しているディスプレイドライバの仕様によるものです。表 1.4 に設定可能な色深度を示します。

表 1.4 LCD のインタフェースごとの色深度対応

インタフェース	ディスプレイドライバ	設定可能な色深度
RGB（パラレルインタフェース）	GUIDRV_LIN	32bpp, 16bpp, 8bpp, 4bpp, 1bpp
SPI（シリアルインタフェース）	GUIDRV_FlexColor	24bpp, 16bpp

1.9　フレームバッファ、ラインバッファ

1.9.1　フレームバッファ（RGB（パラレルインタフェース）を使用する場合）

フレームバッファは LCD に表示する画面の描画データを格納するためのメモリ領域です。emWin により書き込まれ、GLCDC が読み出して LCD に出力します。emWin FIT モジュールではマルチバッファリングに対応しており、デフォルトではダブルバッファ（2 面を交互に切り替え）の設定となっています。

フレームバッファ用の領域は基本的に内蔵 RAM、または拡張 RAM から確保する必要があります。

なお、emWin FIT モジュールではフレームバッファを領域として確保しておらず、コンフィグレーションオプションの「EMWIN_GUI_FRAME_BUFFERx」に内蔵 RAM、または拡張 RAM 上の空き領域の先頭アドレスを指定します。そのため、emWin が使用する作業バッファ（「EMWIN_GUI_NUM_BYTES」でサイズを指定）や B, R、Heap セクション領域などにフレームバッファは含まれておりませんので、これらの使用領域との重複にご注意ください。

また、LCD のサイズと色深度によって必要なフレームバッファのサイズは大きく変動します。コンフィグレーションオプションの「EMWIN_NUM_BUFFERS」を“1”に変更することでシングルバッファも可能ですが、アニメーションのように頻繁な書き換えをするとちらつきが発生するため、ダブルバッファをご使用いただくことを推奨いたします。

フレームバッファのサイズは以下の計算式で求めることができます。

フレームバッファのサイズ(1 面あたり)[bytes] = 1 ラインあたりのバイト数 × LCD の高さ

1 ラインあたりのバイト数は LCD の幅と色深度によって計算方法が変わります。以下に 480px × 272px のサイズを例に色深度ごとの計算方法を説明します。

[色深度：16bpp (2byte) の場合]

1 ラインあたりのバイト数 = 480px × 2byte = 960byte

フレームバッファのサイズ(1 面あたり) = 960byte × 272px = 255Kbyte

ダブルバッファ = 255Kbyte × 2 面 = 510Kbyte

[色深度：8bpp (1byte) の場合]

1 ラインあたりのバイト数 = 480px × 1byte = 480byte ですが、

GLCDC の制限により 1 ラインあたりのバイト数を 64byte で割り切れる値にする必要があり、

480byte / 64 = 7.5 と小数が入る場合は繰り上げが必要となります。

(表示されない余分な余白領域が必要となります。)

これにより、

1 ラインあたりのバイト数 = 64byte × 8 = 512byte

フレームバッファのサイズ(1 面あたり) = 512byte × 272px = 136Kbyte

ダブルバッファ = 136Kbyte × 2 面 = 272Kbyte

[色深度：4bpp (1/2byte) の場合]

1 ラインあたりのバイト数 = 480px × 4bit(1/2byte) = 240byte ですが、

8bpp と同様に 64byte で割り切れる値にする必要があり、

240byte / 64 = 3.75 と小数が入るため繰り上げが必要となります。

これにより、

1 ラインあたりのバイト数 = 64byte×4 = 256byte

フレームバッファのサイズ(1 面あたり) = 256byte×272px = 68Kbyte

ダブルバッファ = 68Kbyte × 2 面 = 136Kbyte

以下に GLCDC が搭載されている RX グループの RAM 容量とフレームバッファの配置例を示します。

表 1.5 GLCDC が搭載されている RX グループの RAM 容量

デバイス	内蔵 RAM	拡張 RAM
RX65N,RX651 グループ	256KB	384KB
RX72N,RX72M,RX66N グループ	512KB	512KB

注. 内蔵 RAM と拡張 RAM のアドレスは不連続のためご注意ください。

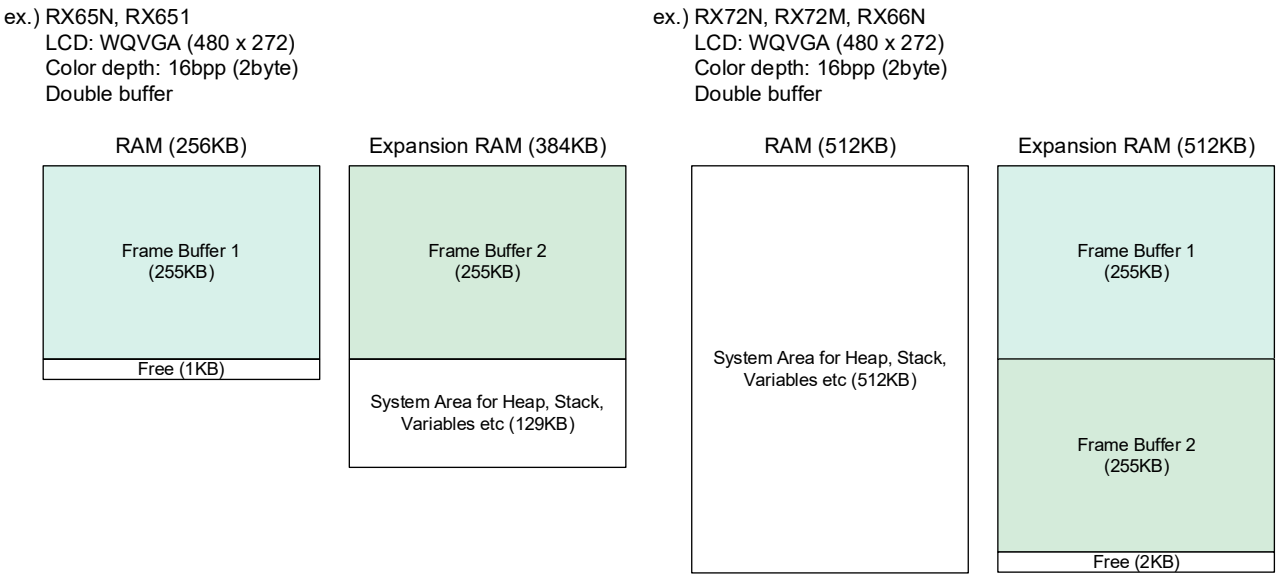


図 1.4 フレームバッファの配置例

1.9.2　ラインバッファ（SPI（シリアルインタフェース）を使用する場合）

ラインバッファは LCD に表示する 1 ライン分の描画データを格納するためのメモリ領域です。emWin により書き込まれ、SPI 通信で 1 ラインずつ LCD にデータが送信されます。

ラインバッファ用の領域は emWin が使用する作業バッファ（「EMWIN_GUI_NUM_BYTES」でサイズを指定）内から確保されます。

ラインバッファのサイズは以下の計算式で求めることができます。

ラインバッファのサイズ[bytes] = LCD の横幅 × (色深度(bpp) / 8)

1.10 使用画像のカラーフォーマットと色深度との関係

emWin では LCD コントローラと色深度に応じたカラーコンバージョンを実行しています。AppWizard や Bitmap Converter で取り込んだ画像を C ソースとして出力する際、基本的にカラーコンバージョンに対応するカラーフォーマットで出力します。カラーコンバージョンに対応するカラーフォーマットを下記表に示します。

透過（alpha）や圧縮（compress）が必要な場合は各々に対応したフォーマットを使用してください。

表 1.6 色深度とカラーフォーマットの設定（RGB（パラレルインタフェース）：GUIDRV_LIN）

色深度	カラーコンバージョン	AppWizard	Bitmap Converter
32	GUICC_M8888I	True color (M8888I) on demand, otherwise high color (M565)	True color with alpha, r/b swapped, alpha inverted
16	GUICC_M565	High color (565), RB swap	High color (565), red and blue swapped
8	GUICC_8666	8bpp	8 bit per pixel (これを選択するためには「Image」→「Convert to」から 8 BPP への変換が必要です)
4	GUICC_16	4bpp	4 bit per pixel (これを選択するためには「Image」→「Convert to」から 4BPP への変換が必要です)
1	GUICC_1	1bpp	1 bit per pixel (これを選択するためには「Image」→「Convert to」から 1BPP への変換が必要です)

表 1.7 色深度とカラーフォーマットの設定（SPI（シリアルインタフェース）：GUIDRV_FlexColor）

色深度	カラーコンバージョン	AppWizard	Bitmap Converter
24	GUICC_888	True color (8888) on demand, otherwise high color (565)	True color 24bpp
16	GUICC_565	High color (565)	High color (565)

2. API 情報

本 FIT モジュールを使用して実装する際には、下記の事項に注意してください。

2.1 ハードウェアの要求

ご使用になる MCU が以下の機能をサポートしている必要があります。

<共通>

- GPIO
- CMT (LCD 用： 1 ch、タッチ用：1 ch)

<SPI（シリアルインタフェース）の LCD を使用する場合>

下記のいずれか

- SCI (1 ch)
- RSPI (1 ch)

下記のいずれか

- DMAC (2 ch)
- DTC

<RGB（パラレルインタフェース）の LCD を使用する場合>

- DMAC (1 ch)
- GLCDC
- DRW2D

<タッチ機能を使用する場合>

- SCI (1 ch)
- RSPI (1 ch)

2.2 ソフトウェアの要求

このドライバは以下の FIT モジュールに依存しています。

<共通>

- ボードサポートパッケージ (r_bsp) Rev.7.60 以降
- GPIO (r_gpio_rx) Rev.5.20 以降
- CMT (r_cmt_rx) Rev.5.72 以降

<SPI（シリアルインタフェース）の LCD を使用する場合>

下記のいずれか

- SCI (r_sci_rx) Rev.5.50 以降
- RSPI (r_rspi_rx) Rev.3.70 以降

下記のいずれか

- DMAC (r_dmaca_rx) Rev.3.42 以降

- DTC (r_dtc_rx) Rev.4.52 以降

<RGB（パラレルインタフェース）の LCD を使用する場合>

- DMAC (r_dmaca_rx) Rev.3.42 以降
- グラフィック LCD コントローラ (r_glcdc_rx) Rev.1.61 以降
- DRW2D ドライバ (r_drw2d_rx) Rev.1.14 以降

<タッチ機能を使用する場合>

下記のいずれか

- SCI(簡易 I²C モード) (r_sci_iic_rx) Rev.2.81 以降
- SCI (r_sci_rx) Rev.5.50 以降
- RSPI (r_rspi_rx) Rev.3.70 以降

2.3　サポートされているツールチェーン

本 FIT モジュールは「8.1 動作確認環境」に示すツールチェーンで動作確認を行っています。

2.4　ヘッダファイル

すべての API 呼び出しとそれをサポートするインタフェース定義は r_emwin_rx_if.h に記載しています。

2.5　整数型

このドライバは ANSI C99 を使用しています。これらの型は stdint.h で定義されています。

2.6　コンパイル時の設定

emWin FIT モジュールのコンフィグレーションオプションの設定は、r_emwin_rx_config.h で行います。
オプション名および設定値に関する説明を、下表に示します。

Configuration options in r_emwin_rx_config.h	
EMWIN_GUI_NUM_BYTES	GUI で使用する最大メモリサイズを設定してください。 必要とされるメモリサイズは開発されるシステムによって変わります。デフォルト値で正常に動作しない場合はサイズを増やしてください。
EMWIN_XSIZE_PHYS	横方向の表示画面サイズを設定してください。
EMWIN_YSIZE_PHYS	縦方向の表示画面サイズを設定してください。
EMWIN_BITS_PER_PIXEL	ピクセル当たりの色深度を選択してください。 “1”の場合、色深度 1bpp が設定されます。この場合、DRW2D を使用することはできません。 “4”の場合、色深度 4bpp が設定されます。この場合、DRW2D を使用することはできません。 “8”の場合、色深度 8bpp が設定されます。この場合、DRW2D を使用することはできません。 “16”の場合、色深度 16bpp が設定されます。この値がデフォルトとなります。 “24”の場合、色深度 24bpp が設定されます。 “32”の場合、色深度 32bpp が設定されます。 LCD のインタフェースにより設定可能な色深度は異なります。詳細は 1.8 色深度を参照してください。
EMWIN_DISPLAY_ORIENTATION	LCD に画像を表示する向き（以後、オリエンテーション）を定義値から選択してください。 “ORIENTATION_0”の場合、画面に正対する向きに表示します。 “ORIENTATION_CW”の場合、時計回りに 90 度回転した向きに表示します、 “ORIENTATION_180”の場合、180 度回転した向きに表示します。 “ORIENTATION_CCW”の場合、反時計回りに 90 度回転した向きに表示します。 色深度が“1”の場合、“ORIENTATION_0”と“ORIENTATION_180”以外は選択できません。 色深度が“4”の場合、“ORIENTATION_0”以外は選択できません。
EMWIN_USE_RUNTIME_ORIENTATION	動作中のオリエンテーション変更機能の有無を選択してください。 “0”の場合、動作中にオリエンテーションを変更できません。 “1”の場合、動作中にオリエンテーションを変更できます。 AppWizard の ROTATEDISPLAY 機能や、LCD_ROTATE_SetSel 関数などを使用した回転に対応します。 “1”で使用する場合、GUI_TOUCH_GetState 関数や GUI_MTOUCH_GetTouchInput 関数で取得した座標の変換が必要になる場合があります。詳細は 6.1.5 動作中のオリエンテーション変更機能を使用する場合のタッチ座標を参照してください。
EMWIN_LCD_IF	LCD に画像を表示するためのインタフェースを選択してください。 “LCD_IF_GLCDC”の場合、GLCDC を使用します。 “LCD_IF_RSPI”の場合、RSPI を使用します。 “LCD_IF_SCI_SPI”の場合、SCI（簡易 SPI モード）を使用します。 “LCD_IF_OTHER”の場合、使用するインタフェースを実装してください。
EMWIN_USE_TOUCH	タッチ機能使用の有無を選択してください。 “0”の場合、タッチ機能を使用しません。 “1”の場合、タッチ機能を使用します。
RGB（パラレルインタフェース）の LCD を使用する場合に有効 (EMWIN_LCD_IF = LCD_IF_GLCDC)	
EMWIN_NUM_BUFFERS	画像の描画に使用するフレームバッファ数を設定してください。emWin FIT モジュールのサポートしている範囲は 1~3 です。それ以上のフレームバッファを使用する場合はフレームバッファ数設定箇所の実装を修正してください。

EMWIN_GUI_FRAME_BUFFER1	画像の描画に使用するフレームバッファ 1 の先頭アドレスを設定してください。
EMWIN_GUI_FRAME_BUFFER2	画像の描画に使用するフレームバッファ 2 の先頭アドレスを設定してください。 EMWIN_NUM_BUFFERS が“2”未満の場合、本設定は無効です。
EMWIN_GUI_FRAME_BUFFER3	画像の描画に使用するフレームバッファ 3 の先頭アドレスを設定してください。 EMWIN_NUM_BUFFERS が“3”未満の場合、本設定は無効です。
EMWIN_USE_DRW2D	DRW2D 使用の有無を選択してください。 “0”の場合、DRW2D を使用しません。 “1”の場合、DRW2D を使用します。
EMWIN_DMAC_NUMBER	フレームバッファ間のデータ転送に使用する DMAC のチャネル番号を設定してください。
EMWIN_INIT_DMAC	emWin FIT モジュール内で DMAC の初期化をするかを設定してください。 “0”の場合、emWin FIT モジュール内で DMAC の初期化を行いません。 “1”の場合、emWin FIT モジュール内で DMAC の初期化を行います。
EMWIN_USE_DISP_SIGNAL_PIN	LCD のリセット制御ピン使用の有無を選択してください。 “0”の場合、LCD のリセット制御ピンを使用しません。 “1”の場合、LCD のリセット制御ピンを使用します。
EMWIN_DISP_SIGNAL_PIN	LCD のリセット制御ピン使用時に、使用する GPIO のピン番号を設定してください。設定には、GPIO FIT モジュールの列挙型 gpio_port_pin_t のメンバを使用してください。
EMWIN_USE_BACKLIGHT_PIN	LCD のバックライト制御ピン使用の有無を選択してください。 “0”の場合、LCD のバックライト制御ピンを使用しません。 “1”の場合、LCD のバックライト制御ピンを使用します。
EMWIN_BACKLIGHT_PIN	LCD のバックライト制御ピン使用時に、使用する GPIO のピン番号を設定してください。 設定には、GPIO FIT モジュールの列挙型 gpio_port_pin_t のメンバを使用してください。
SPI（シリアルインタフェース）の LCD を使用する場合に有効 (EMWIN_LCD_IF = LCD_IF_RSPI or LCD_IF_SCI_SPI)	
EMWIN_LCD_IF_NUMBER	LCD の表示に使用するインタフェースのチャネル番号を設定してください。
EMWIN_LCD_DRIVER_IC	LCD に搭載されている LCD コントローラの型名を定義値から選択してください。 “LCD_DRV_IC_ST7715”の場合、ST7715 シリーズに対応する制御を行います。 “LCD_DRV_IC_ILI9341”の場合、ILI9341 シリーズに対応する制御を行います。 “LCD_DRV_IC_OTHER”の場合、使用する LCD に搭載されている IC の制御コードを実装してください。
EMWIN_LCD_BAUDRATE	LCD の表示に使用するインタフェースのボーレートを設定してください。設定可能な最大値は使用される LCD コントローラや RX 製品によって異なりますので、LCD コントローラのデータシートや RX 製品のユーザーズマニュアルを参照してください。
EMWIN_GUI_USE_CACHE	キャッシュ使用の有無を選択してください。 “0”の場合、キャッシュを使用しません。 “1”の場合、キャッシュを使用します。 キャッシュを使用することで内部処理が高速になります。ただし、キャッシュを使用するためには LCD の画面サイズ分のメモリ容量が必要となりますので、 EMWIN_GUI_NUM_BYTES で 1 フレーム分のメモリを追加で確保してください。 　必要メモリ容量(バイト) : 画面サイズ×色深度 表示メモリの読み出しが不可能な LCD の場合、キャッシュを有効にするか、メモリデバイス機能を使用することで、読み出しせずに描画が可能です。また、色深度が“24”のときにキャッシュは使用できません。その場合は“0”を選択してください。
EMWIN_SELECT_DMAC_DTC	LCD コントローラとのデータ送受信に使用する DMA コントローラを選択してください。 “0”の場合、割り込みを使用します。（DMA コントローラを使用しない） “1”の場合、DTC を使用します。 “2”の場合、DMAC を使用します。

EMWIN_DMAC_NUMBER	LCD コントローラへのデータ送信に使用する DMAC のチャネル番号を設定してください。 データ送受信方法に DTC を選択した場合は無視されます。
EMWIN_DMAC_NUMBER2	LCD コントローラからのデータ受信に使用する DMAC のチャネル番号を設定してください。 データ送受信方法に DTC を選択した場合は無視されます。
EMWIN_INIT_DMAC	emWin FIT モジュール内で DMAC/DTC の初期化をするかを設定してください。 “0”の場合、emWin FIT モジュール内で DMAC/DTC の初期化を行いません。 “1”の場合、emWin FIT モジュール内で DMAC/DTC の初期化を行います。
EMWIN_USE_DISP_SIGNAL_PIN	LCD のリセット制御ピン使用の有無を選択してください。 “0”の場合、LCD のリセット制御ピンを使用しません。 “1”の場合、LCD のリセット制御ピンを使用します。
EMWIN_DISP_SIGNAL_PIN	LCD のリセット制御ピン使用時に、使用する GPIO のピン番号を設定してください。設定には、GPIO FIT モジュールの列挙型 gpio_port_pin_t のメンバを使用してください。
EMWIN_USE_BACKLIGHT_PIN	LCD のバックライト制御ピン使用の有無を選択してください。 “0”の場合、LCD のバックライト制御ピンを使用しません。 “1”の場合、LCD のバックライト制御ピンを使用します。
EMWIN_BACKLIGHT_PIN	LCD のバックライト制御ピン使用時に、使用する GPIO のピン番号を設定してください。設定には、GPIO FIT モジュールの列挙型 gpio_port_pin_t のメンバを使用してください。
EMWIN_USE_DATA_CMD_PIN	LCD のデータ/コマンド制御ピン使用の有無を選択してください。 “0”の場合、LCD のデータ/コマンド制御ピンを使用しません。 “1”の場合、LCD のデータ/コマンド制御ピンを使用します。
EMWIN_DATA_CMD_PIN	LCD のデータ/コマンド制御ピン使用時に、使用する GPIO のピン番号を設定してください。設定には、GPIO FIT モジュールの列挙型 gpio_port_pin_t のメンバを使用してください。
EMWIN_USE_LCD_CS_PIN	LCD の CS 制御ピン使用の有無を選択してください。 “0”の場合、LCD の CS 制御ピンを使用しません。 “1”の場合、LCD の CS 制御ピンを使用します。
EMWIN_LCD_CS_PIN	LCD の CS 制御ピン使用時に、使用する GPIO のピン番号を設定してください。設定には、GPIO FIT モジュールの列挙型 gpio_port_pin_t のメンバを使用してください。
タッチ機能を使用する場合に有効	
EMWIN_TOUCH_IF	タッチ機能で使用するインタフェースを定義値から選択してください。 “TOUCH_IF_SCI_IIC”の場合、SCI-IIC を使用します。 “TOUCH_IF_RSPI”の場合、RSPI を使用します。 “TOUCH_IF_SCI_SPI”の場合、SCI（簡易 SPI モード）を使用します。 “TOUCH_IF_OTHER”の場合、使用するインタフェースを実装してください。 RSPI、または SCI（簡易 SPI モード）を使用する場合に設定が必要です。 詳細は、6.3.3 タッチを使用する場合 を参照してください。
EMWIN_TOUCH_IF_NUMBER	タッチ機能で使用するインタフェースのチャネル番号を設定してください。
EMWIN_TOUCH_BAUDRATE	タッチ機能で使用するインタフェースのボーレートを設定してください。設定可能な最大値は使用されるタッチコントローラや RX 製品によって異なりますので、タッチコントローラのデータシートや RX 製品のユーザーズマニュアルを参照してください。
EMWIN_USE_MULTITOUCH	マルチタッチ機能の使用の有無を選択してください。 “0”の場合、マルチタッチ機能を使用しません。 “1”の場合、マルチタッチ機能を使用します。 タッチ機能で使用するインタフェースに RSPI または SCI（簡易 SPI モード）を選択したときにはマルチタッチ機能を使用できません。また、マルチタッチに対応していないタッチコントローラでも同様に使用できません。その場合は“0”を設定してください。
EMWIN_MAX_NUM_TOUCHPOINTS	マルチタッチ機能使用時に、タッチパネルのポイントの最大数を設定してください。
EMWIN_SLAVE_ADDRESS	タッチパネルのスレーブアドレスを設定してください。 本設定はタッチ機能で使用するインタフェースに I^2C を選択したときに有効です。
EMWIN_USE_TOUCH_IC_RESET_PIN	タッチパネル IC リセットピン使用の有無を選択してください。 “0”の場合、タッチパネル IC リセットピンを使用しません。 “1”の場合、タッチパネル IC リセットピンを使用します。

EMWIN_TOUCH_IC_RESET_PIN	タッチパネル IC リセットピン使用時に、使用する GPIO のピン番号を設定してください。設定には、GPIO FIT モジュールの列挙型 gpio_port_pin_t のメンバを使用してください。
EMWIN_USE_TOUCH_CS_PIN	タッチパネルの CS 制御ピン使用の有無を選択してください。 “0”の場合、タッチパネルの CS 制御ピンを使用しません。 “1”の場合、タッチパネルの CS 制御ピンを使用します。
EMWIN_TOUCH_CS_PIN	タッチパネルの CS 制御ピン使用時に、使用する GPIO のピン番号を設定してください。設定には、GPIO FIT モジュールの列挙型 gpio_port_pin_t のメンバを使用してください。

2.7　コードサイズ

emWin FIT モジュールの ROM サイズ、RAM サイズ、最大使用スタックサイズを下表に示します。

ROM (コードおよび定数) と RAM (グローバルデータ) のサイズは、ビルド時の「2.6 コンパイル時の設定」のコンフィグレーションオプションによって決まります。

下表の値は下記条件で確認しています。

モジュールリビジョン: emWin Rev6.52 FIT Rev.1.00

コンパイラバージョン: Renesas Electronics C/C++ Compiler Package for RX Family V3.07.00
(統合開発環境のデフォルト設定に”-lang = c99”オプションを追加)
GCC for Renesas RX 14.02.00.202505
(統合開発環境のデフォルト設定に” -std=gnu99”オプションを追加)
IAR C/C++ Compiler for Renesas RX version 5.20.1
(統合開発環境のデフォルト設定)

コンフィグレーションオプション: デフォルト設定

ROM、RAM およびスタックのコードサイズ

デバイス	分類	使用メモリ		
		Renesas Compiler	GCC	IAR Compiler
RX130	ROM	151210 バイト	148936 バイト	36730 バイト
	RAM	5388 バイト	5418 バイト	3944 バイト
	スタック	700 バイト	ー	ー
[コンフィグレーションオプション] EMWIN_GUI_NUM_BYTES = 4 EMWIN_LCD_IF = LCD_IF_RSPI EMWIN_SELECT_DMAC_DTC = 1 EMWIN_USE_TOUCH = 1 EMWIN_TOUCH_IF = TOUCH_IF_SCI_SPI				
RX231	ROM	149351 バイト	156979 バイト	29576 バイト
	RAM	3792 バイト	3822 バイト	1478 バイト
	スタック	700 バイト	ー	ー
[コンフィグレーションオプション] EMWIN_GUI_NUM_BYTES=4 EMWIN_LCD_IF = LCD_IF_SCI_SPI EMWIN_SELECT_DMAC_DTC = 2 EMWIN_USE_TOUCH = 0				
RX65N	ROM	179339 バイト	179168 バイト	38431 バイト
	RAM	4149 バイト	4182 バイト	3138 バイト
	スタック	1064 バイト	ー	ー
[コンフィグレーションオプション] EMWIN_GUI_NUM_BYTES=4 EMWIN_LCD_IF = LCD_IF_GLCDC EMWIN_USE_DRW2D = 1 EMWIN_USE_TOUCH = 1 EMWIN_TOUCH_IF = TOUCH_IF_SCI_IIC EMWIN_USE_MULTITOUCH = 0				

ROM、RAM およびスタックのコードサイズ				
デバイス	分類	使用メモリ		
		Renesas Compiler	GCC	IAR Compiler
RX72N	ROM	179355 バイト	179174 バイト	38523 バイト
	RAM	4149 バイト	4182 バイト	3138 バイト
	スタック	1024 バイト	－	－
[コンフィグレーションオプション] EMWIN_GUI_NUM_BYTES=4 EMWIN_LCD_IF = LCD_IF_GLCDC EMWIN_USE_DRW2D = 1 EMWIN_USE_TOUCH = 1 EMWIN_TOUCH_IF = TOUCH_IF_SCI_IIC EMWIN_USE_MULTITOUCH = 0				

RAM サイズには EMWIN_GUI_NUM_BYTES の値が含まれます。本表に記載の RAM サイズは EMWIN_GUI_NUM_BYTES に 4（最小値）を設定したときのサイズです。

2.8　引数

API 関数の引数である構造体を示します。この構造体は、API 関数のプロトタイプ宣言とともに r_emwin_rx_if.h に記載されています。

2.9　FIT モジュールの追加方法

emWin FIT モジュールは、使用するプロジェクトごとに追加する必要があります。ルネサスでは、スマート・コンフィグレータを使用した(1)(2)(3)の追加方法を推奨しています。ただし、CS+および IAREW 上では QE for Display[RX]との連携はできませんので、ユーザにて設定を行っていただく必要があります。

(1) e² studio 上でスマート・コンフィグレータを使用して FIT モジュールを追加する場合
e² studio のスマート・コンフィグレータを使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX スマート・コンフィグレータ ユーザーガイド: e² studio 編 (R20AN0451)」を参照してください。

※ダウンロードした FIT モジュールの保存先に、他のバージョンの emWin FIT モジュールが存在する場合、適切にモジュールを追加することができないことがあります。該当のフォルダ内には最新版の emWin FIT モジュールのみを格納してください。

(2) CS+上でスマート・コンフィグレータを使用して FIT モジュールを追加する場合
CS+上で、スタンドアロン版スマート・コンフィグレータを使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX スマート・コンフィグレータ ユーザーガイド: CS+編 (R20AN0470)」を参照してください。

(3) IAREW 上でスマート・コンフィグレータを使用して FIT モジュールを追加する場合
スタンドアロン版スマート・コンフィグレータを使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX スマート・コンフィグレータ ユーザーガイド: IAREW 編 (R20AN0535)」を参照してください。

2.10　for 文、while 文、do while 文について

本モジュールでは、レジスタの反映待ち処理などで for 文、while 文、do while 文（ループ処理）を使用しています。これらループ処理には、「WAIT_LOOP」をキーワードとしたコメントを記述しています。そのため、ループ処理にユーザがフェイルセーフの処理を組み込む場合は「WAIT_LOOP」で該当の処理を検索できます。

以下に記述例を示します。

```
while 文の例：
/* WAIT_LOOP */
while(0 == SYSTEM.OSCOVFSR.BIT.PLOVF)
{
   /* The delay period needed is to make sure that the PLL has stabilized. */
}

for 文の例：
/* Initialize reference counters to 0. */
/* WAIT_LOOP */
for (i = 0; i < BSP_REG_PROTECT_TOTAL_ITEMS; i++)
{
   g_protect_counters[i] = 0;
}

do while 文の例：
/* Reset completion waiting */
do
{
   reg = phy_read(ether_channel, PHY_REG_CONTROL);
   count++;
} while ((reg & PHY_CONTROL_RESET) && (count < ETHER_CFG_PHY_DELAY_RESET)); /* WAIT_LOOP */
```

3. emWin から呼び出される関数

3.1 GUI_X_Config()

この関数は、emWin のメモリ管理システムで使用するメモリブロックを登録する関数です。

Format

void GUI_X_Config(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

emWin のメモリ管理システムで使用するメモリブロックの登録に使用されます。

emWin FIT モジュールでは、GUI ブロックの関数を使用してメモリを割り当てています、

Reentrant

- 不可

3.2　LCD_X_Config ()

この関数は、LCD とデバイスドライバを初期化する関数です。

Format

void LCD_X_Config(void)

Parameters

なし

Return Values

なし

Properties

LCD.h にプロトタイプ宣言されています。

Description

LCD とデバイスドライバの初期化に使用されます。

emWin FIT モジュールでは、GUI ブロックの関数を使用して LCD を初期化します。インタフェースに GLCDC を選択した場合、併せて DRW2D FIT モジュールも初期化しています。

Reentrant

- 不可

3.3 LCD_X_DisplayDriver ()

この関数は、ディスプレイドライバのコールバック関数です。

Format

```
int LCD_X_DisplayDriver(
        unsigned layer_index,
        unsigned cmd,
        void * p_data
        )
```

Parameters

layer_index	入力	レイヤ番号
cmd	入力	実行するコマンド
p_data	入力	データ構造へのポインタ

Return Values

0:	コマンドが正常に実行された
-1:	コマンドが実行されなかった
-2:	エラー発生

Properties

LCD.h にプロトタイプ宣言されています。

Description

ディスプレイドライバのコールバック関数として使用されます。ディスプレイドライバから呼び出され、コールバックルーチンを実行します。

emWin FIT モジュールでは、コマンドに応じてインタフェースとして選択した各周辺機能の初期化やインタフェースに応じた処理を行っています。

インタフェースに GLCDC を選択した場合、GLCDC FIT モジュールの初期化と DRW2D FIT モジュールを使用した図形生成関数の登録、Lookup Table エントリの設定、ディスプレイの ON/OFF、バッファの切り替えを行っています。

インタフェースに RSPI や SCI（簡易 SPI モード）を選択した場合、RSPI FIT モジュールまたは SCI FIT モジュールの初期化と、ディスプレイの ON/OFF を行っています。

コマンド	値	意味	対応状況 〇：対応 ×：未対応
LCD_X_INITCONTROLLER	0x01	ディスプレイコントローラの初期化	〇
LCD_X_SETVRAMADDR	0x02	Video RAM アドレスの設定	×
LCD_X_SETORG	0x03	レイヤ内の基準の設定	×
LCD_X_SETLUTENTRY	0x04	Lookup Table のエントリの設定	〇
LCD_X_ON	0x05	ディスプレイのスイッチオン	〇
LCD_X_OFF	0x06	ディスプレイのスイッチオフ	〇
LCD_X_SETSIZE	0x07	レイヤサイズの設定	×
LCD_X_SETPOS	0x08	レイヤ位置の設定	×
LCD_X_SETVIS	0x09	レイヤの可視化の設定	×
LCD_X_SETALPHA	0x0A	レイヤのアルファ値の設定	×
LCD_X_SETALPHAMODE	0x0B	アルファブレンディングモードの設定	×
LCD_X_SETCHROMAMODE	0x0C	クロマブレンディングモードの設定	×
LCD_X_SETCHROMA	0x0D	クロマ値の設定	×
LCD_X_SHOWBUFFER	0x0E	バッファの切り替え	〇

Reentrant

- 不可

3.4　GUI_X_Init ()

この関数は、GUI に必要なハードウェアを初期化する関数です。

Format

void GUI_X_Init(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

必要なハードウェアを初期化する関数です。

emWin FIT モジュールでは、待ち時間の計測に使用するためのコンペアマッチタイマの初期化に使用されています。

Reentrant

- 不可

3.5　GUI_X_Delay ()

この関数は、指定の時間を待つ関数です。

Format

```
void GUI_X_Delay(
        int ms
        )
```

Parameters

ms	入力	待ち時間[ミリ秒]

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

指定の時間を待ちます。

emWin FIT モジュールでは、コンペアマッチタイマから得られる時間情報を利用して、指定した時間を待ちます。

Reentrant

- 不可

3.6　GUI_X_ExecIdle ()

この関数は、Window Manager から GUI が最新の状態のため処理すべき内容がないときに呼び出される関数です。

Format

void GUI_X_ExecIdle(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

Window Manager から、GUI が最新の状態で処理すべき内容がないときに呼び出されます。

emWin FIT モジュールでは、何も処理をしません。

Reentrant

- 不可

3.7　GUI_X_GetTime ()

この関数は、現在のシステム時間がミリ秒単位の整数型で得られる関数です。

Format

```
GUI_TIMER_TIME GUI_X_GetTime(
                    int ms
                    )
```

Parameters

なし

Return Values

システム時間[ミリ秒]

Properties

GUI.h にプロトタイプ宣言されています。

Description

現在のシステム時間がミリ秒単位の整数型で得られます。

emWin FIT モジュールでは、コンペアマッチタイマから取得される値を返しています。

Reentrant

- 不可

3.8　GUI_X_ErrorOut ()

この関数は、致命的なエラーが発生した際に、エラー文字列を入力として emWin から呼び出される関数です。

Format

```
void GUI_X_ErrorOut(
        const char *s
        )
```

Parameters

s	入力	エラー文字列

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

致命的なエラーが発生した際に、エラー文字列を入力として emWin から呼び出されます。

GUI_DEBUG_LEVEL ≧ 3 で有効となります。

emWin FIT モジュールでは、何も処理をしません。

Reentrant

- 不可

3.9　GUI_X_Warn ()

この関数は、警告が発生した際に、警告文字列を入力として emWin から呼び出される関数です。

Format

```
void GUI_X_Warn(
        const char *s
        )
```

Parameters

s	入力	警告文字列

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

警告が発生した際に、警告文字列を入力として emWin から呼び出されます。

GUI_DEBUG_LEVEL ≧ 4 で有効となります。

emWin FIT モジュールでは、何も処理をしません。

Reentrant

- 不可

3.10　GUI_X_Log ()

この関数は、メッセージが発生した際に、メッセージ文字列を入力として emWin から呼び出される関数です。

Format

```
void GUI_X_Log(
        const char *s
        )
```

Parameters

s	入力	メッセージ文字列

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

メッセージが発生した際に、メッセージ文字列を入力として emWin から呼び出されます。

GUI_DEBUG_LEVEL ≧ 5 で有効となります。

emWin FIT モジュールでは、何も処理をしません。

Reentrant

- 不可

3.11　GUI_X_InitOS ()

この関数は、マルチタスク環境で使用する場合に、セマフォまたは mutex を生成する関数です。

Format
void GUI_X_InitOS(void)

Parameters
なし

Return Values
なし

Properties
GUI.h にプロトタイプ宣言されています。

Description
マルチタスク環境で使用する場合に、セマフォまたは mutex を生成する関数です。

emWin FIT モジュールでは、FreeRTOS 使用時は、FreeRTOS の関数を使用して、セマフォの生成とイベントの生成を行っています。FreeRTOS 不使用時は、何も処理をしません。

Reentrant
- 不可

3.12　GUI_X_Unlock ()

この関数は、マルチタスク環境で使用する場合に、GUI をアンロックする関数です。

Format

void GUI_X_Unlock(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

マルチタスク環境で使用する場合に、GUI をアンロックする関数です。

emWin FIT モジュールでは、FreeRTOS 使用時は、FreeRTOS の関数を使用して、セマフォを解放しています。FreeRTOS 不使用時は、何も処理をしません。

Reentrant

- 不可

3.13　GUI_X_Lock ()

この関数は、マルチタスク環境で使用する場合に、GUI をロックする関数です。

Format

void GUI_X_Unlock(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

マルチタスク環境で使用する場合に、GUI をロックする関数です。

emWin FIT モジュールでは、FreeRTOS 使用時は、FreeRTOS の関数を使用して、セマフォを取得しています。FreeRTOS 不使用時は、何も処理をしません。

Reentrant

- 不可

3.14　GUI_X_GetTaskId ()

この関数は、マルチタスク環境で使用する場合に、タスク ID を取得する関数です。

Format

U32 GUI_X_GetTaskId(void)

Parameters

なし

Return Values

タスク ID

Properties

GUI.h にプロトタイプ宣言されています。

Description

マルチタスク環境で使用する場合に、タスク ID を取得する関数です。

emWin FIT モジュールでは、FreeRTOS 使用時は、FreeRTOS の関数を使用して、タスクハンドルを取得しています。FreeRTOS 不使用時は、固定で 1 を返しています。

Reentrant

- 不可

3.15　GUI_X_WaitEvent ()

この関数は、マルチタスク環境で使用する場合に、イベント待ちする関数です。

Format

void GUI_X_WaitEvent(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

マルチタスク環境で使用する場合に、イベント待ちする関数です。

emWin FIT モジュールでは、FreeRTOS 使用時は、FreeRTOS の関数を使用してイベント待ちを行っています。ここで、最大待ち時間は 60000 ミリ秒としています。FreeRTOS 不使用時は、何も処理をしません。

Reentrant

- 不可

3.16　GUI_X_SignalEvent ()

この関数は、マルチタスク環境で使用する場合に、イベント通知する関数です。

Format

void GUI_X_SignalEvent(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

マルチタスク環境で使用する場合に、イベント通知する関数です。

emWin FIT モジュールでは、FreeRTOS 使用時は、FreeRTOS の関数を使用してイベントを通知しています。FreeRTOS 不使用時は、何も処理をしません。

Reentrant

- 不可

3.17　GUI_X_WaitEventTimed ()

この関数は、マルチタスク環境で使用する場合に、指定した期間のイベント待ちする関数です。

Format

```
void GUI_X_WaitEventTimed(
        int period
        )
```

Parameters

Period	入力	指定期間

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

マルチタスク環境で使用する場合に、指定した期間のイベント待ちする関数です。

emWin FIT モジュールでは、FreeRTOS 使用時は、FreeRTOS の関数を使用して、指定した期間のイベント待ちを行っています。ここで、最大待ち時間は 60000 ミリ秒としています。FreeRTOS 不使用時は、何も処理をしません。

Reentrant

- 不可

3.18　PID_X_SetLayerIndex ()

この関数は、レイヤ番号を設定する関数です。

Format

```
void PID_X_SetLayerIndex(
        int layer_index
        )
```

Parameters

LayerIndex	入力	レイヤ番号

Return Values

なし

Properties

PIDConf.h にプロトタイプ宣言されています。

Description

レイヤ番号を設定します。

emWin FIT モジュールでは、内部変数にレイヤ番号を設定します。

Reentrant

- 不可

3.19　PID_X_Init ()

この関数は、Pointer Input Device を初期化する関数です。

Format

void PID_X_Init(void)

Parameters

なし

Return Values

なし

Properties

PIDConf.h にプロトタイプ宣言されています。

Description

Pointer Input Device の初期化を行います。

emWin FIT モジュールでは、Touch IC のリセット、タッチに使用する周辺機能の初期化、タッチ情報を取得するためのコンペアマッチタイマ起動とコールバック関数登録、マルチタッチ機能の有効化を行っています。

Reentrant

- 不可

3.20　GUI_TOUCH_X_ActiveX ()

この関数は、Touch IC の X 軸の電圧測定を有効にする関数です。

Format

void GUI_TOUCH_X_ActivateX(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

Touch IC の X 軸の電圧測定を有効にする関数です。

emWin FIT モジュールでは、何も処理をしません。

Reentrant

- 不可

3.21　GUI_TOUCH_X_ActiveY ()

この関数は、Touch IC の Y 軸の電圧測定を有効にする関数です。

Format

void GUI_TOUCH_X_ActivateY(void)

Parameters

なし

Return Values

なし

Properties

GUI.h にプロトタイプ宣言されています。

Description

Touch IC の Y 軸の電圧測定を有効にする関数です。

emWin FIT モジュールでは、何も処理をしません。

Reentrant

- 不可

3.22　GUI_TOUCH_X_MeasureX ()

この関数は、Touch IC から得られる X 軸の電圧測定結果を返す関数です。

Format

int GUI_TOUCH_X_MeasureX(void)

Parameters

なし

Return Values

0

Properties

GUI.h にプロトタイプ宣言されています。

Description

Touch IC から得られる X 軸の電圧測定結果を返す関数です。

emWin FIT モジュールでは、固定で 0 を返します。

Reentrant

- 不可

3.23　GUI_TOUCH_Y_MeasureY ()

この関数は、Touch IC から得られる Y 軸の電圧測定結果を返す関数です。

Format

int GUI_TOUCH_X_MeasureY(void)

Parameters

なし

Return Values

0

Properties

GUI.h にプロトタイプ宣言されています。

Description

Touch IC から得られる Y 軸の電圧測定結果を返す関数です。

emWin FIT モジュールでは、固定で 0 を返します。

Reentrant

- 不可

3.24　APPW_X_FS_Init ()

この関数は、AppWizard のファイルシステムアクセスを初期化する関数です。

Format

void APPW_X_FS_Init (void)

Parameters

なし

Return Values

なし

Properties

AppWizard.h にプロトタイプ宣言されています。

Description

AppWizard のファイルシステムアクセスを初期化する関数です。

emWin FIT モジュールでは、何も処理をしません。

Reentrant

- 不可

4. アプリケーションから呼び出される API 関数

4.1 R_EMWIN_GetBufferAddr()

この関数は、emWin FIT モジュールで使用されるフレームバッファのアドレスを取得する関数です。

Format

void * R_EMWIN_GetBufferAddr (void)

Parameters

なし

Return Values

フレームバッファ・アドレス

Properties

r_emwin_rx_if.h にプロトタイプ宣言されています。

Description

emWin FIT モジュールで使用されるフレームバッファのアドレスを取得します。

Reentrant

- 不可

4.2　R_EMWIN_GetD2 ()

この関数は、emWin FIT モジュールの Dave2D 機能のハンドルを取得する関数です。

Format

d2_device * R_EMWIN_GetD2 (void)

Parameters

なし

Return Values

Dave2D のハンドル

Properties

r_emwin_rx_if.h にプロトタイプ宣言されています。

Description

emWin FIT モジュールの Dave2D 機能のハンドルを取得します。

本関数は、DRW2D FIT モジュール使用時のみ有効になります。

Reentrant

- 不可

4.3　R_EMWIN_EnableDave2D ()

この関数は、emWin FIT モジュールの Dave2D 機能を動作許可状態にする関数です。

Format

void R_EMWIN_EnableDave2D (void)

Parameters

なし

Return Values

なし

Properties

r_emwin_rx_if.h にプロトタイプ宣言されています。

Description

emWin FIT モジュールの Dave2D 機能を動作許可状態にします。

本関数は、DRW2D FIT モジュール使用時のみ有効になります。

Reentrant

- 不可

4.4　R_EMWIN_DisableDave2D ()

この関数は、emWin FIT モジュールの Dave2D 機能を動作禁止状態にする関数です。

Format

void R_EMWIN_DisableDave2D (void)

Parameters

なし

Return Values

なし

Properties

r_emwin_rx_if.h にプロトタイプ宣言されています。

Description

emWin FIT モジュールの Dave2D 機能を動作禁止状態にします。

本関数は、DRW2D FIT モジュール使用時のみ有効になります。

Reentrant

- 不可

4.5　R_EMWIN_GetDaveActive ()

この関数は、emWin FIT モジュールの Dave2D 機能の動作状態を取得する関数です。

Format

uint32_t R_EMWIN_GetDaveActive (void)

Parameters

なし

Return Values

Dave2D 動作状態（0:動作禁止状態、1:動作許可状態）

Properties

r_emwin_rx_if.h にプロトタイプ宣言されています。

Description

emWin FIT モジュールの Dave2D 機能の動作状態を取得します。

本関数は、DRW2D FIT モジュール使用時のみ有効になります。

Reentrant

- 不可

4.6　R_EMWIN_GetVersion ()

この関数は、emWin FIT モジュールのバージョン番号を取得する関数です。

Format

void R_EMWIN_GetVersion(st_emwin_version_t * version)

Parameters

* version　　出力　　バージョン番号の格納先ポインタ

Return Values

なし

Properties

r_emwin_rx_if.h にプロトタイプ宣言されています。

Description

emWin FIT モジュールのバージョン番号を取得します。

Reentrant

- 不可

4.7　_VSYNC_ISR ()

この関数は、V-sync 割り込みの処理を行う関数です。

Format

void _VSYNC_ISR(void * p)

Parameters

* p	出力	GLCDC からの Callback argument

Return Values

なし

Properties

r_emwin_rx_if.h にプロトタイプ宣言されています。

Description

V-sync 割り込みの処理を行います。

GLCDC FIT モジュールのコールバック関数を想定しています。

Reentrant

- 不可

5. 端子設定

emWin FIT モジュールを使用するための端子設定を QE for Display[RX]で行うことができます。

設定が必要な端子は、LCD パネルのリセット端子、LCD パネルのバックライト端子、LCD パネル搭載のタッチ IC のリセット端子などです。RGB や SPI 接続の LCD に合わせて使用する端子を選択してください。

e^2 studio の場合、QE for Display[RX]の emWin の設定ダイアログの端子設定機能を使用することで端子設定を行うことができます。QE for Display[RX]を使用する場合、r_emwin_rx に関するスマート・コンフィグレータによる端子の設定は不要です。

選択した端子情報は qe_emwin_config.h に出力され、2.6 コンパイル時の設定に示すマクロ定義の値が設定されます。その際に、r_emwin_rx_config.h のマクロ定義は無効化されます。

QE for Display[RX]を使用せずに端子設定を行う場合、emWin FIT モジュールに含まれる r_emwin_rx_config.h を編集してください。

6. 実装上の注意事項

本 FIT モジュールを使用して実装する際には、下記の事項に注意してください。

6.1 共通事項

6.1.1 ライブラリファイルの選択

emWin FIT モジュールには、以下のライブラリファイルが含まれています。使用する MCU とコンパイラに応じたライブラリファイルを選択して使用してください。なお、スマート・コンフィグレータを使用した場合はデバイスと使用するコンパイラに応じて自動的にライブラリが設定されます。

表 6.1-1　ライブラリの構成

ライブラリファイル	
emWinLib_RXv1_CCRX.lib	Renesas Electronics C/C++ Compiler for RX Family で使用するライブラリファイル。 _d.lib : 倍精度浮動小数演算処理対応ライブラリ _s.lib : 単精度浮動小数演算処理対応ライブラリ
emWinLib_RXv2_CCRX.lib	
emWinLib_RXv3_CCRX_d.lib	
emWinLib_RXv3_CCRX_s.lib	
libemWinLib_RXv1_GCC.a	GCC for Renesas RX で使用するライブラリファイル。
libemWinLib_RXv2_GCC.a	
libemWinLib_RXv3_GCC.a	
emWinLib_RXv1_IAR.a	IAR C/C++ Compiler for Renesas RX で使用するライブラリファイル。 _d.a : 倍精度浮動小数演算処理対応ライブラリ _s.a : 単精度浮動小数演算処理対応ライブラリ
emWinLib_RXv2_IAR.a	
emWinLib_RXv3_IAR_d.a	
emWinLib_RXv3_IAR_s.a	

6.1.2 システムの動作に必要な RAM サイズの設定

emWin FIT モジュールは、システムで占める RAM の割合が多く、開発されるシステムよっても必要な RAM サイズは変わります。そのため、RX 製品によってデフォルトのサイズではシステムが正常に動作しない場合があります。

必要に応じて以下のサイズを調整してください。設定はスマート・コンフィグレータから行うことができます。

r_bsp の設定項目 (スマート・コンフィグレータから設定可能)

- ユーザスタック (BSP_CFG_USTACK_BYTES)
- 割り込みスタック (BSP_CFG_ISTACK_BYTES)
- ヒープサイズ (BSP_CFG_HEAP_BYTES)

emWin FIT モジュールの設定項目(スマート・コンフィグレータ、または QE for Display[RX] から設定可能)

- GUI で使用する最大メモリサイズ (EMWIN_GUI_NUM_BYTES)

GUI で使用する最大メモリサイズ (EMWIN_GUI_NUM_BYTES)の設定値を決める手段として、GUI_ALLOC_GetMemInfo 関数が活用できます。システムの動作をひととおり操作してから本関数を実行す

ると、システムの動作で使用されたメモリ使用量などの情報を取得できます。詳細は以下のドキュメントを参照してください。

・emWin Graphic Library with Graphical User Interface User Guide & Reference Manual

（https://www.segger.com/downloads/emwin/UM03001）

6.1.3　画像の形式

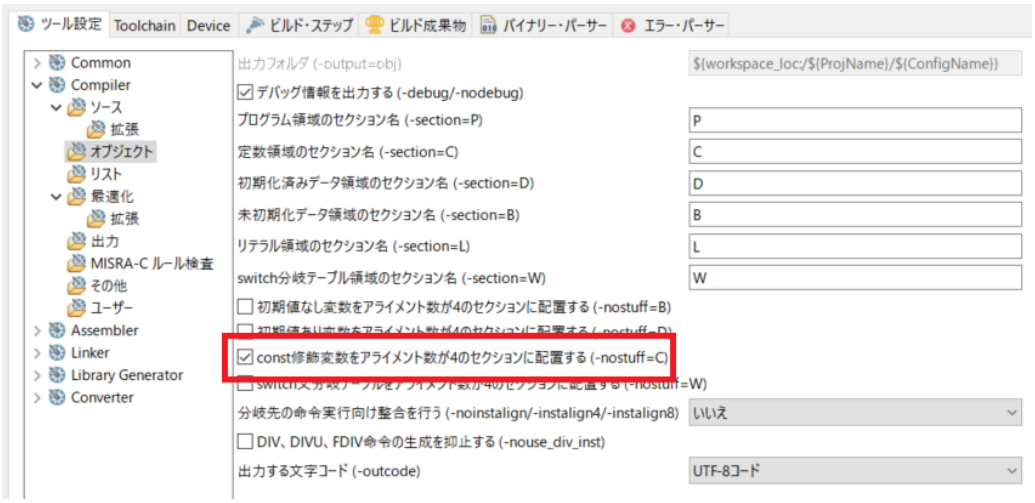
emWin FIT モジュールで画像を使用する場合はビットマップ形式(.bmp)のデータを使用してください。

6.1.4　データのアライメントの設定

画像やフォントのデータを 4 の倍数のアドレス(4 バイトアライメント)に配置する必要があります。

<CC-RX の場合>

e² studio で[プロジェクト]→[C/C++ Project Settings]でプロパティ画面を開き、[C/C++ビルド]→[設定]の[ツール設定]タブを開き、[Compiler]→[オブジェクト]の“const 修飾変数をアライメント数が 4 のセクションに配置する(-nostuff=C)”の項目にチェックを入れてください。なお、スマート・コンフィグレータを使用した場合は自動的に設定されます。



<GCC, IAR の場合>

GCC, IAR では画像データの各変数に対して個別にアライメントを指定する必要があります。CC-RX のようにまとめて指定する方法はありません。ただし、emWin FIT モジュールに同梱している GUI デザインツール（AppWizard や Bitmap Converter）から出力されるソースコードの画像データの各変数には GUI_CONST_STORAGE マクロが付加されています。このマクロの検索および置換などの方法で各変数に対してアライメントを指定してください。

GCC: __attribute__ ((aligned(4)))

IAR: #pragma data_alignment=4

6.1.5　動作中のオリエンテーション変更機能を使用する場合のタッチ座標

EMWIN_USE_RUNTIME_ORIENTATION を“1”に設定して使用する場合、GUI_TOUCH_GetState 関数や GUI_MTOUCH_GetTouchInput 関数で取得したタッチ座標は動作中のオリエンテーションに応じて変換が必要です。

1.　NHD-4.3-480272EF-ATXL#-CTP(Newheaven Display)や ER-TFT043-3(East Rising)の場合
表 6.1-2 に座標の変換方法を示します。

表 6.1-2　タッチ座標の変換方法（1）

動作中の画面の向き	変換方法 (番号順に変換処理してください)
ORIENTATION_0	変換不要
ORIENTATION_CW	1. X 座標の反転 (x = (画面の幅 – 1) – x) 2. XY 座標のスワップ (x ⇔ y)
ORIENTATION_180	1. X 座標の反転 (x = (画面の幅 – 1) – x) 2. Y 座標の反転 (y = (画面の高さ – 1) – y)
ORIENTATION_CCW	1. Y 座標の反転 (x = (画面の高さ – 1) – y) 2. XY 座標のスワップ (x ⇔ y)

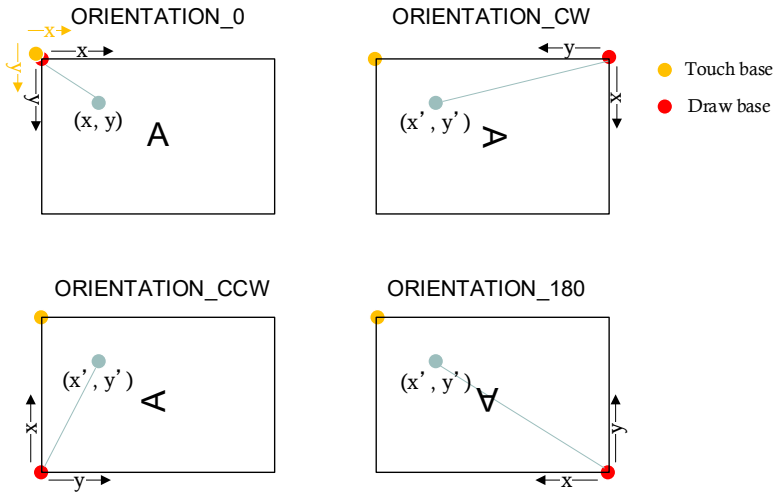


図 6.1.1 オリエンテーションごとの座標

2. MSP2807 の LCD の場合

MSP2807 は基点が画面右上となるため、NHD-4.3-480272EF-ATXL#-CTP(Newheaven Display)や ER-TFT043-3(East Rising)と異なり、画面左上を基点にする変換も併せて必要です。表 6.1-3 に座標の変換方法を示します。

表 6.1-3　タッチ座標の変換方法（2）

動作中の画面の向き	変換内容
ORIENTATION_0	1. X 座標の反転 (x = (画面の幅 – 1) – x)
ORIENTATION_CW	1. XY 座標のスワップ (x ⇔ y)
ORIENTATION_180	1. Y 座標の反転 (y = (画面の高さ – 1) – y)
ORIENTATION_CCW	1. Y 座標の反転 (x = (画面の高さ – 1) – y) 2. XY 座標のスワップ (x ⇔ y)

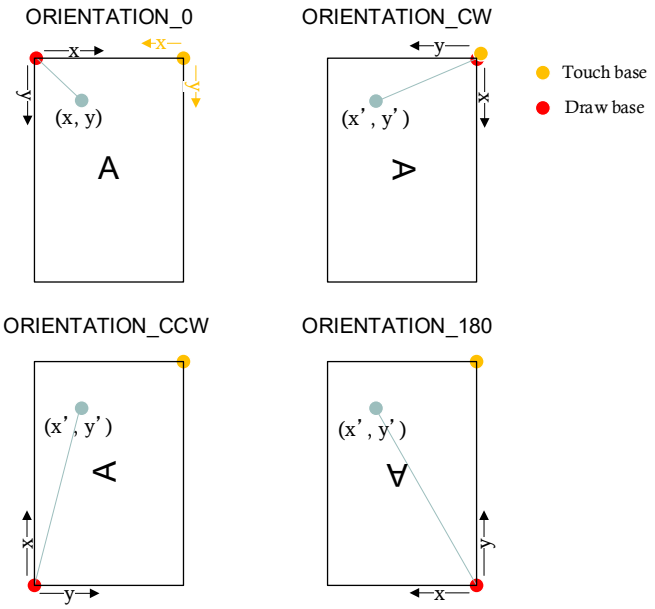


図 6.1.2 オリエンテーションごとの座標

6.1.6　AppWizard とその他ツールのご使用について

emWin FIT モジュールには emWin で開発するための多くのツールを同梱しています。その中の一つに AppWizard があります。AppWizard はこれ一つで画面のデザインから画像やフォントの処理、アニメーションやモーション制御、インタラクションの設定などを網羅した統合管理ツールです。AppWizard でデザインするとこれらを容易に組み込める反面、RAM/ROM の使用量や処理負荷が高くなります。そのため、デバイスの性能によっては AppWizard によるデザイン手法が適さない場合があります。特に RAM/ROM のサイズが小さい製品では影響が大きくなります。

開発に着手いただく前に十分検証していただくことを推奨いたします。

表 6.1-4　インタフェースとデザインツール

インタフェース	デバイス	開発
RGB （パラレルインタフェース）	GLCDC 搭載した RX 製品	AppWizard または、GUI Builder などその他ツールで開発可能です。
SPI （シリアルインタフェース）	RX600, RX700 シリーズ RX100, RX200 シリーズ	AppWizard で開発可能です。RAM/ROM 使用量や処理速度を重視する場合は GUI Builder やその他のツールでの開発を推奨します。

6.1.7　DMAC/DTC のご使用について

emWin FIT モジュールでは DMAC/DTC の制御に FIT モジュールを使用しています。FIT モジュールとコード生成コンポーネントの DMAC/DTC は併用できません。ユーザプログラムで DMAC/DTC を制御する場合は FIT モジュールを使用してください。

6.1.8　各周辺機能の割り込み優先レベルについて

emWin FIT モジュールで使用する各周辺機能の割り込み優先レベル（デフォルト）は以下のとおりです。ユーザシステムに合わせて設定してください。

表 6.1-5　emWin FIT で使用する周辺モジュールの割り込み優先レベルル

周辺機能	設定箇所	優先レベル
CMT	コンフィグレーション	5
GLCDC	コンフィグレーション	5
SCI(簡易 SPI)	ソースファイル内 （LCDConf_spi_if.c、r_emwin_rx_pid_spi_if.c）	5
RSPI	コンフィグレーション	3
DMAC	ソースファイル内（LCDConf_spi_if.c）	10
SCI(簡易 I^2C)	コンフィグレーション	2

6.2　RGB（パラレルインタフェース）の LCD を使用する場合

6.2.1　RX65N を使用する場合のセクション設定

emWin FIT モジュールでは、インタフェースとして GLCDC を使用する場合にフレームバッファを 2 つ確保する必要があります。RX65N で使用する場合には、アドレス配置の関係より、フレームバッファを 2 ヶ所に分けて配置する必要があります。そのため、フレームバッファを 0x00000100 から 256Kbyte、0x00800000 から 256Kbyte を確保した場合、元から設定されている SU 以降のセクションは、0x00840000 以降に設定してください。

6.2.2　DRW2D 有効時のヒープサイズの設定

DRW2D を有効にする場合、スマート・コンフィグレータを使用して「r_bsp」の「Heap size」を 0x4000 に変更してください。なお、0x4000 は目安であり、必要とされるヒープサイズは開発されるシステムによって変わります。

6.2.3　AppWizard 使用時のマルチバッファリングの設定

AppWizard で設定したウィジェットをちらつきなくスムーズに動作させるためにはマルチバッファリングオプションを有効にする必要があります。

弊社の環境と AppWizard を組み合わせて使用する都合上、AppWizard のプロジェクトのプロパティにある「Selected BSP」の項目を「None」にする必要があります。その場合、マルチバッファリングのオプション「Enable Multibuffering」はデフォルトで無効になっていますので、有効に設定してください。

また、QE for Display[RX] V3.01.00 以降のバージョンでは、QE for Display[RX]から AppWizard を起動するとき、マルチバッファリングを有効にして AppWizard のプロジェクトファイルを生成しますので、マルチバッファリングを有効にする作業は不要です。

6.3　SPI（シリアルインタフェース）の LCD を使用する場合

6.3.1　ちらつきを軽減する方法について

SPI（シリアルインタフェース）の LCD を使用する場合、デフォルト設定のまま使用いただくと LCD 表示にちらつきが発生します。ちらつきなくスムーズに動作させるためには以下の方法を使用してください。

- キャッシュの有効　（1 フレーム分のバッファが必要です）
- メモリデバイス機能[注]の使用（ユーザアプリケーションプログラムでの設定が必要です）

注. メモリデバイス機能は emWin に実装されている描画操作で使われる一時的なバッファおよび様々な操作ができる機能です。詳細は emWin のユーザガイドを参照してください。

6.3.2　色深度設定について

emWin FIT モジュールは下記の LCD コントローラに対応しています。これらのコントローラの色深度は最大 18 ビットのため、EMWIN_BITS_PER_PIXEL で“24”(RGB888 の 24 ビット)を指定した場合、18 ビットに減色されます。

- ST7715 シリーズ
- ILI9341 シリーズ

6.3.3　タッチを使用する場合

SCI FIT モジュールまたは RSPI FIT モジュールの設定において、リード動作時の送信ダミーデータの設定を“0x00”に設定してください。

- SCI FIT モジュールの場合：SCI_CFG_DUMMY_TX_BYTE
- RSPI FIT モジュールの場合：RSPI_CFG_DUMMY_TXDATA

6.3.4　データ送受信に DTC/DMAC を使用する場合

LCD への表示データの送受信に DTC/DMAC を使用する場合、2.6 コンパイル時の設定 の設定に加えて、各 FIT モジュールの設定が必要です。詳細は SCI FIT モジュール、RSPI FIT モジュール、DTC FIT モジュールや DMAC FIT モジュールのマニュアルを参照してください。

6.3.5　高速通信における通信端子の駆動能力の設定

1 MHz 以上の高速通信を行う場合、基板や配線の寄生成分などによって通信波形が乱れて正常に通信できない場合があります。そのため、通信に使用する出力端子の駆動能力を「高駆動出力」や「高速インタフェース用高駆動出力」の設定にすることを推奨いたします。詳細は RX 製品のユーザーズマニュアルの I/O ポート章を参照してください。

6.4　動作確認環境以外での使用

emWin FIT モジュールを、動作確認環境に示されている環境以外で使用する際には、以下の事項に注意して使用してください。動作確認環境は 8.1 章を参照してください。

- LCD の使用

emWin FIT モジュールでは、LCD とのインタフェースの設定に以下の方法を使用することができます。

1. QE for Display[RX]を使用した設定

   e² studio においてスマート・コンフィグレータを使用して emWin FIT モジュールを追加した後に、QE for Display[RX]において必要な設定を入力してください。詳細は以下の URL を参照してください。

   ディスプレイ対応開発支援ツール QE for Display | Renesas

2. スマート・コンフィグレータによる設定

   プロジェクト作成時にスマート・コンフィグレータを使用し、ソフトウェアコンポーネント設定画面において必要な設定を入力してください。

   ※QE for Display[RX]を使用した場合、QE for Display[RX]の設定がスマート・コンフィグレータで設定した項目より優先されます。

3. 設定データ構造体の実装（LCD とのインタフェースに GLCDC を選択した場合）

   スマート・コンフィグレータ/QE for Display を使用せず、LCDConf_glcdc_if.c ファイル内の r_emwin_lcd_open 関数において、R_GLCDC_Open 関数を使用している箇所で設定データ構造体を実装してください。

4. マクロ定義やソースコードの編集（LCD とのインタフェースに RSPI、SCI（簡易 SPI モード）を選択した場合）

   LCDConf_rspi_if.c、LCDConf_sci_spi_if.c ファイル内に SPI 接続の LCD に応じたマクロ定義とソースコードがあります。8.1 動作確認環境 に記載していない LCD の動作は未確認のため、既存の設定やソースコードで正常に動作しない場合はご使用の LCD に合わせて変更が必要です。また、r_emwin_rx_config.h で EMWIN_LCD_DRIVER_IC に LCD_DRV_IC_OTHER を選択した場合、ご使用される LCD に応じたマクロ定義やソースコードを実装してください。e² studio を使用して実装する場合、実装する必要のある個所に warning が表示されます。

また、GLCDC FIT モジュールや RSPI FIT モジュール、SCI FIT モジュールのいずれも使用しない場合は LCDConf_user_if.c 内に処理を実装してください。e² studio を使用して実装する場合、r_emwin_rx_config.h の EMWIN_LCD_IF に LCD_IF_OTHER を設定することで、インタフェースを実装する必要のある箇所に warning が表示されます。

- タッチパネルの使用

emWin FIT モジュールでは、I²C と SPI インタフェースで使用する場合の処理が実装されています。他のタッチパネル、他のインタフェースを使用する場合には、以下に必要な処理を実装してください。

1. PIDConf.c ファイル内の、タッチデータを取得し emWin ライブラリに渡している部分（pidconf_cb_single 関数）
2. r_emwin_rx_pid_user_if.c のタッチパネルとのインタフェースを担っている部分

2 については、e² studio を使用して実装する場合、r_emwin_rx_config.h の EMWIN_TOUCH_IF に TOUCH_IF_OTHER を設定することで、ファイル内のコードが有効になります。

- OS の使用

emWin FIT モジュールでは、FreeRTOS を使用する場合（BSP_CFG_RTOS_USED == 1）、または OS を使用しない場合（BSP_CFG_RTOS_USED == 0）をサポートしています。これ以外の OS を使用する場合には、GUI_X_Ex.c 内の OS システムコールを使用している部分に、必要な処理を実装してください。

e² studio を使用して実装する場合、BSP_CFG_RTOS_USED が上記の値以外であれば、OS システムコールを実装する必要のある箇所に warning が表示されます。詳細は、以下のドキュメントを参照してください。

ボードサポートパッケージモジュール Firmware Integration Technology (R01AN1685)

- emWin ライブラリの設定

emWin FIT モジュールでは、emWin ライブラリへの設定の内、フレームバッファ数は 3 面まで、ディスプレイドライバとしては GUIDRV_Lin をサポートしています。表示のちらつきを抑えるためにフレームバッファ数を増やす場合、また上記以外のディスプレイドライバを適用する場合には、以下の部分に必要な処理を実装してください。

1. フレームバッファ数について、4 面以上を使用する場合には、r_emwin_rx_config.h の EMWIN_NUM_BUFFERS に 16 を超えない範囲で使用するバッファ数を指定して、必要な実装を追加してください。e² studio を使用して実装する場合、EMWIN_NUM_BUFFERS に 4 以上の値を設定することで、LCDConf_glcdc_if.c 内の追加実装が必要な箇所に warning が表示されます。
2. ディスプレイドライバについて、GUIDRV_Lin や GUIDRV_FlexColor 以外のディスプレイドライバを使用する場合には、Segger とのライセンス契約を結び、使用するディスプレイドライバが含まれた emWin ライブラリのソースコードを入手いただく必要があります。その後、LCDConf_user_if.c 内に処理を実装してください。e² studio を使用して実装する場合、r_emwin_rx_config.h の EMWIN_DISPLAY_DRIVER に GUIDRV_OTHER を設定することで、ファイル内のコードが有効になります。なお、GUIDRV_Lin や GUIDRV_FlexColor 以外のディスプレイドライバは弊社ではサポート対象外となりますのでご了承ください。

7. サンプルアプリケーション

doc/Training フォルダ内に、サンプルアプリケーションが格納されています。詳細は r_emwin_rx¥doc¥Training フォルダにある以下のドキュメントを参照してください。

- emWin Training（emWin_Training.pdf）

また、Segger 社からも以下の URL に各種 API を使用したサンプルプログラムが紹介されておりますのでご参照ください。

- emWin Examples (https://wiki.segger.com/emWin_Examples)

8. 付録

8.1 動作確認環境

本 FIT モジュールの動作確認環境を以下に示します。

表 8.1 動作確認環境

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e^2 studio 2025-10
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family(CC-RX) V3.07.00 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99 -nostuff=C -head=math
	GCC for Renesas RX 14.02.00.202505 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -std=gnu99
	IAR C/C++ Compiler for Renesas RX version 5.20.1 コンパイルオプション：統合開発環境のデフォルト設定
エンディアン	リトルエンディアン
モジュールのバージョン	Ver.1.00
使用 OS	FreeRTOS Release Release RX MCUs FreeRTOS v1.0.10 comes from original 10.4.3 · renesas/FreeRTOS-Kernel · GitHub
	OS 未使用
使用ボード	Renesas Envision KIT RPBRX65N（型名：RTK5RX65N2C00000BR） Renesas Envision Kit RPBRX72N（型名：RTK5RX72N0C00000BJ） Renesas Starter Kit+ for RX65N-2MB（型名：RTK50565N2S10010BE） Renesas Starter Kit+ for RX72N（型名：RTK5572NNHS10000BE） Renesas Starter Kit for RX660（型名：RTK556609HS00000BE） Renesas Starter Kit for RX140（型名：RTK551406BS00000BE） Renesas Starter Kit for RX231（型名：R0K505231S900BE） Target Board for RX130（型名：RTK5RX1300C00000BR） Target Board for RX671（型名：RTK5RX6710C00000BJ） RX261 MCU グループ評価キット（型名：RTK5EK2610S00001BE）

表 8.2　動作確認 LCD

LCD 型番	解像度	LCD コントローラ	タッチ コントローラ	備考
RGB インタフェース				
NHD-4.3-480272EF-ATXL#-CTP (Newheaven Display)	480×272	HX8257-A (Himax)	FT5306 (FocalTech)	Renesas Starter Kit+ for RX65N-2MB に搭載 Renesas Starter Kit+ for RX72N に搭載
ER-TFT043-3 (East Rising)	480×272	NV3047 (TDK)	FT5206 (FocalTech)	Renesas Envision KIT RPBRX65N に搭載 Renesas Envision Kit RPBRX72N に搭載
シリアルインタフェース				
RH128128T-1x44WN-B2 (OKAYA)	128×128	ST7715R (Sitronix)	—	Renesas Starter Kit 付属 Pmod 接続 LCD
MSP2807 (Kuongshun Electronic Limited)	240×320	ILI9341 (ILITEK)	XPT2046 (Xptek)	動作確認に使用

9. 参考ドキュメント

ユーザーズマニュアル：ソフトウェア

・emWin Wiki

(https://wiki.segger.com/emWin)

・emWin Graphic Library with Graphical User Interface User Guide & Reference Manual

(https://www.segger.com/doc/UM03001_emWin.html) オンライン版

（https://www.segger.com/downloads/emwin/UM03001）PDF 版

・AppWizard User Guide & Reference Manual

(https://www.segger.com/doc/UM03003_AppWizard.html) オンライン版

ユーザーズマニュアル：ハードウェア

（各デバイスの最新版をルネサス エレクトロニクスホームページから入手してください）

テクニカルアップデート／テクニカルニュース

（最新の情報をルネサスエレクトロニクスホームページから入手してください）

ユーザーズマニュアル：開発環境

RX ファミリ CC-RX コンパイラ ユーザーズマニュアル（R20UT3248）

（最新版をルネサス エレクトロニクスホームページから入手してください）

テクニカルアップデートの対応について

本モジュールにはテクニカルアップデートはありません。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Dec.9.25	—	新規発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。
5. クロックについて
   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
8. 製品間の相違について
   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

   標準水準：　コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等

   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.5.0-1　2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。

© 2025 Renesas Electronics Corporation. All rights reserved.