RX600 シリーズ

M3S-GUI-LIB: GUI ライブラリ導入ガイド

R20AN0077JJ0101
Rev.1.01
2011.06.20

要旨

本資料では、RX600 シリーズ用 M3S-GUI-LIB(以降 GUI ライブラリ)の使用方法とサンプルプログラムについて記載しています。

GUI ライブラリは、M3S-GUI-BUILDER(以降 GUI ビルダ)を用いて作成したプログラムを各種マイコン上で実行するためのライブラリです。GUI ビルダの導入ガイドを先に参照ください。

GUI ビルダおよび GUI ライブラリはルネサスの web サイトよりダウンロードできます。

動作確認デバイス

RX600 シリーズ

目次

1. 製品構成

本製品は、以下のものから構成されています。

- M3S-GUI-LIB V.2.01 Release00
- M3S-GUI-LIB V.2.01 Release00 導入ガイド（本書：r20an0077jj0101_rx_gui.pdf）

本製品の型名は「R0MRX60GL0011RRC」です。

本製品をインストールするには、setup.exe を実行してください。またインストール時に表示される使用許諾契約書に同意いただく必要があります。

インストールすると以下のデータがコピーされます。

表 1.1 製品構成

		内容
インストーラ(setup.exe)		Windows 用のインストーラです。 表示される使用許諾契約書に同意いただいた場合、以下フォルダにデータがコピーされます。 C:¥Renesas¥an_r20an0077jj_rx_gui_v201r00
ドキュメント(doc)		
	r20an0077jj0101_rx_gui.pdf	導入ガイド
GUI ライブラリ(lib)		
	mgt_rx_little.lib mgt_rx_big.lib	GUI ライブラリ
ヘッダファイル(include)		
	mgt.h	GUI ライブラリ用ヘッダファイル
	ximg.h	プラグイン用ヘッダファイル
	font_sample_mgt.h	サンプルフォントライブラリ用ヘッダファイル
グラフィックライブラリ(lib_ext)		
	rx600lewgp16.lib rx600bewgp16.lib	M3S-GRAPHIC-LIB(以降グラフィックライブラリ) 無償評価版
	font_sample_rx_little.lib, font_sample_rx_big.lib	サンプルフォントデータ
サンプルデータ(sample)		
	WQVGA_Design_sample	Visual C#のプロジェクト
	GUI_RX62N_RSK_sample	RN62N-RSK 用開発環境

本製品に含まれるライブラリのバージョンは以下のとおりです。

表 1.2 バージョン

名称	バージョン
GUI ライブラリ	1.1.1.01

2. 仕様

本 GUI ライブラリは、GUI ビルダの仕様に準拠しています。

Win32 版は状態遷移情報をログファイルに出力できますが、マイコン版はログ出力機能には対応していません。詳しくは GUI ビルダのユーザーズマニュアルを参照ください。

2.1 開発環境

以下の開発環境で動作します。

[統合開発環境]

High Performance Embedded Workshop Version 4.09.00.007 以降

[C コンパイラ]

C/C++ compiler package for RX family V.1.01 Release 00 以降

ライブラリは以下のオプション（デフォルト）で生成されています。

[リトルエンディアン]

-cpu=rx600 -output=obj="$(CONFIGDIR)¥$(FILELEAF).obj" -nologo

[ビッグエンディアン]

上記に加えて -endian=big を設定

2.2 ROM/RAM/スタックサイズ

GUI ライブラリが使用する ROM/RAM/スタックサイズは以下のとおりです。

表 2.1 ROM/RAM サイズ

分類	サイズ
GUI ライブラリの RAM（セクション B,R）	約 450byte
GUI ライブラリの ROM（セクション P,C,D,L）	約 7kB
ROM (GUI ライブラリ + グラフィックライブラリ)	約 16kB
サンプルフォントデータ(1byte フォント)	約 3.2kB
サンプルフォントデータ(2byte フォント)	約 256kB

表 2.2 スタックサイズ

API 関数名	スタックサイズ [byte] (グラフィックライブラリ込み)
mgt_init	8
mgt_flip	16
mgt_use_fontset	4
mgt_set_rdb	20
mgt_tick	472
mgt_reset_focus	32
mgt_transit	8
mgt_hide_control	8
mgt_show_control	8
mgt_invalidate	24
mgt_get_current_form	4
mgt_get_clientdata	8
mgt_get_image	56
mgt_get_checkbox_value	8
mgt_set_checkbox_value	32
mgt_set_button_image	32
mgt_set_form_image	32
mgt_set_picturebox_image	32
mgt_set_listbox_data	32
mgt_set_label_text	32
mgt_make_bitmap	8
_ximg_init	4
_ximg_set_plugin	4

【注】 付属のサンプルプログラムの値です。併用するグラフィックライブラリによってスタックサイズは変化しますので、ユーザは CallWalker 等のスタック算出ツールを使用し、スタックサイズの確認を行ってください。

2.3 バージョン情報

GUI ライブラリは、文字列でバージョン情報を格納しています。以下の extern 宣言によりこの変数にアクセスすることが出来ます。

宣言:

```
extern const uint8_t R_gui_lib_version[];
```

本製品のライブラリに格納されているデータは以下の通りです。

リトルエンディアン:

"M3S-GUI-LIB version 1.1.1.01 for RX LITTLE endian.(Jun 13 2011, 15:17:31)"

ビッグエンディアン:

"M3S-GUI-LIB version 1.1.1.01 for RX BIG endian.(Jun 13 2011, 15:17:34)"

3.　GUIライブラリの使用方法

GUI ビルダで作成したプログラムを、RX 上で動作させるには本 GUI ライブラリが必要です。また使用するハードウェア構成にあわせて、表示機器(LCD など)や入力機器(タッチパネルなど)の制御ドライバを用意してください。

GUI ライブラリは、グラフィックライブラリを用いて表示デバイスのフレームバッファに表示する画像を書き込みます。ライブラリ周辺は以下のような構成になります。

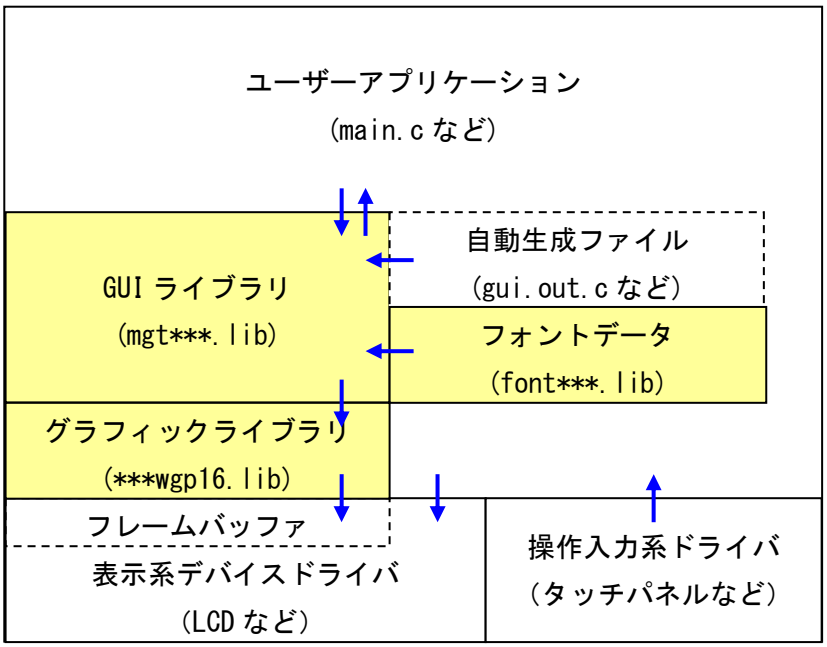

図 3.1 ライブラリ周辺の構成

GUI ビルダで自動生成したファイル(gui.out.c など)と一緒に、以下のライブラリをプロジェクトに組み込んでください。

リトルエンディアンの場合

mgt_rx_little.lib font_sample_rx_little.lib rx600lewgp16.lib

ビッグエンディアンの場合

mgt_rx_big.lib font_sample_rx_big.lib rx600bewgp16.lib

ハードウェア依存の部分を含むファイル(main.c など)を除き、GUI ビルダで生成した GUI 関連のファイルは各マイコンで流用することができます。

4.　サンプルの使用方法

本サンプルプログラムは、RX600 Series Direct Drive LCD Demonstration Application Note Rev.1.01(以降、LCD Direct Drive)のサンプルプログラムをベースに、GUI ライブラリを組み込んでいます。また、M3S-JPEGD-LIB (以降、JPEG デコーダ)とのプラグインを実装し、GUI ライブラリから JPEG データの読み出しを実現しています。

LCD Direct Drive は、以下のハードウェア環境で動作します。

- RX62N TFT-LCD Direct-Drive Demo Kit

各製品に関する情報は、以下のルネサスの web サイトを参照ください。

製品	ウェブページ
RX62N TFT-LCD Direct-Drive Demo Kit	http://www.renesas.eu/products/tools/introductory_evaluation_tools/renesas_starter_kits/rx62n_tftlcd_dddemokit/rx62n_tftlcd_dddemokit.jsp
LCD Direct Drive	http://www.renesas.com/products/mpumcu/rx/Application_Notes.jsp
JPEG デコーダ	http://japan.renesas.com/mw/jpeglib

4.1　サンプルプログラム概要

サンプルプログラムは、2 種類の実装について説明します。

サンプル 1 : 画像情報を外部メモリから読み出す(デフォルト)

サンプル 2 : 画像情報を内蔵 ROM に置いて読み出す

このサンプルプログラムは、「USE_FETCH_IMAGE」マクロの定義により変更できます。

「USE_FETCH_IMAGE」を定義した場合、「GUI_MEMORY_ACCESS_TYPE」を定義して外部メモリから画像を読み出す方法を選択することができます。

また「GUI_PLUGIN_JPEG」を定義した場合、JPEG プラグインを使用することができます。

4.2　サンプルプログラム実行手順

ここでは、サンプルプログラムの実行手順について説明します。

4.2.1　Visual C#プロジェクトのビルド

この章では Visual C#プロジェクトで作成した GUI デザインを GUI プログラム生成ツールを使用してマイコンで使用できる C 言語ソースを出力するまでの手順を説明します。事前に GUI デザインを変更する場合は WQVGA_Design_sample.sln をダブルクリックして Visual C#を起動してください。ただし、GUI デザインを変更した場合は RX62N のサンプルプログラム編集が必要となる場合がありますので注意してください。

(1) GUI ビルダをインストールしたディレクトリの下にある「bin」ディレクトリ一式を WQVGA_Design_sample.sln ファイルと同じ階層にコピーします。

(2)Windows のコマンドプロンプトを起動します。

(3)WQVGA_Design_sample フォルダをカレントに指定し、「make.bat」を実行します。カレントフォルダに gui.out*が出力されることを確認してください。

- Makefile のデフォルトでは guigen のオプションで jpg 画像をビットマップに変換せず出力するように設定にしています。ビットマップに変換する場合は、Makefile を開いて、「--jpg」を削除してください。

(4)続けてコマンドプロンプトから「make.bat copy」を実行します。カレントフォルダにある gui.out.*が DirectLCD プロジェクトの auto_gen ディレクトリにコピーされていることを確認してください。

4.2.2 RX62Nのサンプルプログラムのビルド

ここでは RX62N のサンプルプログラムのビルドから実行までを説明します。サンプルプログラムには画像情報を内蔵 ROM に配置する方法と、外部メモリから読み出す方法を実装しています。それぞれの手順について説明します。

- サンプル 1 : 画像情報を外部メモリから読み出す場合の手順(デフォルト)
  (1) GUI_RX62N_RSK_sample¥ DirectLCD.hws を起動します。
  (2)プロジェクト「r_Packages」をアクティブ（デフォルトでアクティブ状態）とし、LCD Direct Drive のライブラリをビルドしてください。ライブラリのほかにサンプルプログラムに必要なヘッダがインストールディレクトリにあわせて自動生成されます。
  (3) プロジェクト「guiimg」をアクティブにしてビルドしてください。ワークスペースディレクトリに画像情報をバイナリ化した「guiimg.bin」が出力されます。guiimg.bin は外部メモリから画像情報を読み出して動作させる場合に使用します。
  (4) プロジェクト「DirectLCD」をアクティブにします。ビルド環境設定で「USE_FETCH_IMAGE」マクロを有効（デフォルトで有効）にし、サンプルプログラムをビルドしてください。ビルドすると「0 Errors, 5 Warnings」となりますが、オリジナルから無効にした部分のセクションに関するもの、またはツールチェインのバージョンの変更に伴うもので、動作上は問題ありません。
  (5) RX62N-RSK と E1 または E20 を接続してプログラムを書き込みます。
  (6)初回プログラム実行時のみ、エミュレータのコマンドラインウインドウにある「バッチファイルを実行」ボタンを押してください。「guiimg.bin」を SDRAM に展開しシリアルフラッシュに書き込んでプログラムが実行します。「バッチファイルを実行」ボタンが無効である場合は、
  $(PROJDIR)¥ResourceLoad.hdc を指定して実行してください。
  (7)2 回目以降のプログラム実行時は「リセット後実行」ボタンを押してください。

- サンプル 2 : 画像情報を内蔵 ROM に配置する場合の手順
  (1) 「画像情報を外部メモリから読み出す場合の手順」の(1)(2)を実行してください。
  (2) プロジェクト「DirectLCD」をアクティブにします。ビルド環境設定で「USE_FETCH_IMAGE」マクロを無効にした後、サンプルプログラムをビルドしてください。ビルドすると「0 Errors, 5 Warnings」となりますが、オリジナルから無効にした部分のセクションに関するもの、またはツールチェインのバージョンの変更に伴うもので、動作上は問題ありません。
  (3) RX62N-RSK と E1 または E20 を接続してプログラムを書き込み実行してください。
  (4) 「リセット後実行」ボタンを押してプログラムを実行してください。

4.3　サンプルプログラムの説明

4.3.1　画像デコーダとのプラグインインタフェース

GUI ライブラリには JPEG/GIF/PNG デコーダをプラグインすることができます。サンプルプログラムでは JPEG デコーダをプラグインに設定し、JPG 画像を扱えるように実装しています。プラグインの実装方法は r_gui_plugin_jpeg.c または GUI_PLUGIN_JPEG マクロで有効になっている箇所を参照してください。

プラグインを使用しない場合は、GUI_PLUGIN_JPEG マクロを無効にしてください。

4.3.2　外部メモリ上の画像情報の読み出し方法

本サンプルプログラムでは GUI ライブラリから要求される画像データを外部メモリ(サンプルプログラムではシリアルフラッシュ)から読み出し、SDRAM 上の「Resources」変数に格納します。読み出し方法は「GUI_MEMORY_ACCESS_TYPE」マクロの値により 3 種類の読み出し方法を実装しています。環境に合わせて読み出し方法と Resources 変数のサイズを調整してください。また、他の外部メモリを使用する場合はサンプルプログラムの r_gui_fetch_img_flash.c を手本にユーザにて実装してください。

GUI_MEMORY_ACCESS_TYPE のデフォルトは 2、Resources 領域は共通で 2Mbyte 確保しています。

本章では、「GUI_MEMORY_ACCESS_TYPE」について説明します。

「GUI_MEMORY_ACCESS_TYPE」は、「Resources」変数に対して 3 種類の割り当てを持ちます。

- ケース 1 : Resources を静的メモリとして使用し、すべての画像を展開する方法
  ケース 1 ではすべての画像を Resources に展開し、常に保持しておくため、GUI ライブラリが速く画像を読み出せるメリットがありますが、Resources 変数はすべての画像分の容量を確保しなければならないデメリットがあります。
- ケース 2 : Resources を再利用可能な動的メモリとして扱い、すべての画像を展開する方法
  ケース 2 では Resources に読み出した画像を再利用できるメリットがありますが、Resources 変数はテンポラリ領域として多く確保しなければならないデメリットがあります。
- ケース 3 : Resources を再利用不可能な動的メモリとして扱い、画像を一枚だけ展開する方法
  ケース 3 では Resources を少ない容量で使用できるメリットがありますが、GUI ライブラリの画像の読み出しが遅いデメリットがあります。

Resources のデータ構造を以下に示します。

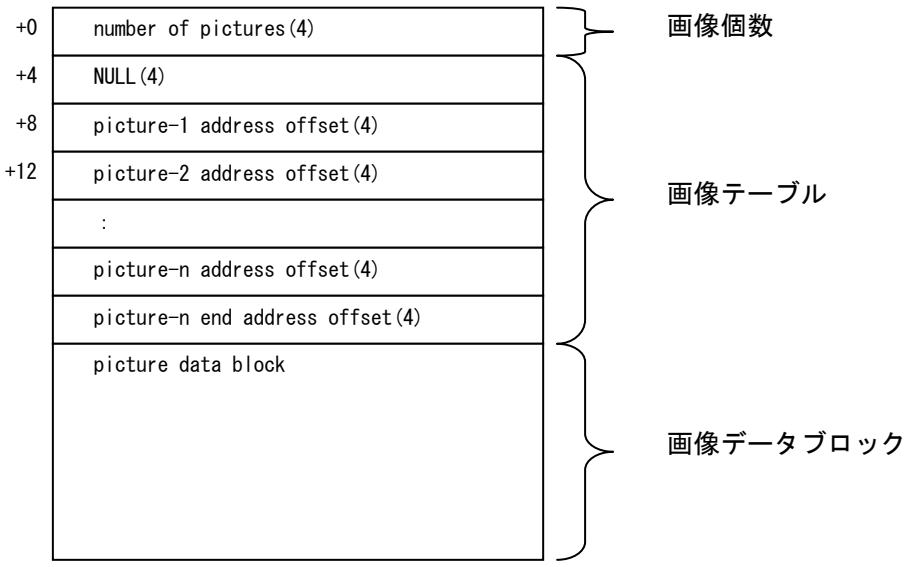

図 4.1 Resources のデータ構造

以降は、画像データブロックの使い方について説明します。

- ケース 1 : Resources を静的メモリとして使用し、すべての画像を展開する方法
  - GUI_MEMORY_ACCESS_TYPE は 0 を定義します。
  - 初回の GUI ライブラリによる画像要求時にすべての画像を展開します。
  - 以降、サンプルプログラムは外部メモリから読み出しません。そのため、Resources は全ての画像サイズ分の容量を必要とします。

- ケース 2 : Resources を再利用可能な動的メモリとして扱い、すべての画像を展開する方法
  - GUI_MEMORY_ACCESS_TYPE は 1 を定義します。
  - GUI ライブラリが最初に画像データを要求したタイミングで、サンプルプログラムは先に画像個数と画像テーブルを読み出します。その後、画像データブロックをキャッシュ領域として読み出した画像を保存します。
  - GUI ライブラリから同じ画像を要求された場合、関数(IMAGE_CACHE_CLEAR())を実行するまでキャッシュ領域が利用できるため、外部メモリへのアクセスは不要です。キャッシュ領域は画面の状態が遷移したときにクリアされます。

- ケース 3 : Resources を再利用不可能な動的メモリとして扱い、画像を一枚だけ展開する方法
  - GUI_MEMORY_ACCESS_TYPE は 2 を定義します。
  - GUI ライブラリが最初に画像データを要求したタイミングで、サンプルプログラムは先に画像個数と画像テーブルを読み出します。その後, サンプルプログラムは画像データブロックを使い捨て領域とピクチャーボックス用キャッシュ領域に分けて読み出した画像を振り分けます。
  - ユーザはピクチャーボックスを設定する前に IMAGE_CACHE_REQ()マクロを実行してください。
  - ピクチャーボックス用キャッシュ領域は画面の状態が遷移したときにクリアされます。
  - ユーザは使い捨て領域とピクチャーボックス用キャッシュ領域の境界を CACHE_INDEX( )マクロで調整することが出来ます。

4.4　LCD Direct Driveサンプルプログラムからの変更内容

LCD Direct Drive のサンプルプログラムからの主な変更内容は、以下のようになります。

- RX ツールチェインのバージョンを 1.1.0.0 に変更
- 使用しないファイルをプロジェクトから除外(ワークスペースのツリーを参照ください)
- GUI 画像データのバイナリファイルを生成するプロジェクト「guiimg」を追加
- GUI 関連のファイル、ライブラリを追加
- GUI 関連のインクルードパスを追加
- DirectLCD プロジェクトのビルド設定に GUI 関連のマクロ定義を追加
- DirectLCD プロジェクトのビルド設定にて L セクションを追加
- EventMgr.c を変更し GUI のメイン処理、外部メモリに画像情報を書き込む処理を追加
- ビルドフェーズの「ResourceBuild」を除外
- config_r_ddlcd.h を変更しドライバ（LCD パネルの表示方向）を変更
- touchscreen.c でディレータスクの時間を修正
- ResourceLoad.hdc を変更し、guiimg.bin を指定
- r_packages プロジェクトのビルド設定を修正

GUI 関連の追加ファイルは以下のとおりです。

表 4.1 GUI 関連の追加ファイル

ディレクトリ	ファイル名	内容
¥GuiBuilder	r_gui_main.c	GUI 関連の初期化、メイン処理など
	r_gui_user.c	GUI 関連のユーザ定義関数や動作処理
	r_gui_fetch_img_sflash.c	フラッシュメモリからの画像取得処理
	r_gui_ipeg_plugin.c	JPEG デコーダとのプラグイン処理
	r_gui_fetch_img.h	外部メモリからの画像取得用ヘッダファイル
¥GuiBuilder¥auto_gen	gui.out.c	自動生成ファイル
	gui.out.h	自動生成ファイル
	gui.out.img0001 ～ 0017	自動生成ファイル
	gui.out.img.c	自動生成ファイル
¥GuiBuilder¥include	mgt.h	GUI ライブラリ用ヘッダファイル
	ximg.h	プラグイン用ヘッダファイル
	font_sample_mgt.h	サンプルフォント用ヘッダファイル
	r_expand_jpegd.h	JPEG デコーダ用ヘッダファイル
¥GuiBuilder¥lib	mgt_rx_little.lib	GUI ライブラリ
	font_sample_rx_little.lib	サンプルフォントデータ
	rx600lewgp16.lib	グラフィックライブラリ
	jpegd_rx_little.lib	JPEG デコーダライブラリ
	expand_jpegd_rx_little.lib	JPEG 伸張ライブラリ

【注】 ディレクトリは"$(WORKSPDIR)"からの相対パスです。

5. 注意事項

- LCD Direct Drive のサンプルプログラムは、RX ツールチェイン V.1.0.2.0 以降でビルド時にエラーが出力されます。本製品に付属しているサンプルプログラムでは暫定対策を実施しています。LCD Direct Drive は必要に応じて最新版に更新してください。
- LCD Direct Drive のライブラリをビルドするとき、HDD が暗号化されている環境では Windows がクラッシュする場合がありますので、問題が発生する場合は暗号化されていない環境でご使用ください。

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://japan.renesas.com/

お問合せ先
http://japan.renesas.com/inquiry

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2011.04.20	—	初版発行
1.01	2011.06.20	—	GUI ライブラリ V.2.01 に合わせてリリース

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. 未使用端子の処理
   【注意】未使用端子は、本文の「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。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】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. リザーブアドレス（予約領域）のアクセス禁止
   【注意】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、将来の機能拡張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。
4. クロックについて
   【注意】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。
5. 製品間の相違について
   【注意】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、内部 ROM、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、ノイズ輻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. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。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。
3. 当社製品を改造、改変、複製等しないでください。
4. 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。お客様の機器の設計において、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には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、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に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   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

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障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おりますが、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、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害など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、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、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。お客様が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. 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固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

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その総株主の議決権の過半数を直接または間接に保有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。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル） (03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/inquiry

Colophon 1.0