

RL78 ファミリ

FPB ボードでスタンドアロン版 QE を用いたタッチアプリケーション開発

要旨

本アプリケーションノートでは、タッチ電極搭載の RL78/G22 FPB (Fast Prototyping Board)を使用し、 静電容量タッチセンシングを応用するアプリケーションの作成に必要な手順について説明します。

本アプリケーションノートは、"CS+、スタンドアロン版スマート・コンフィグレータ、スタンドアロン版 QE for Capacitive Touch"を用いた開発ガイドです。スタンドアロン版 QE for Capacitive Touch ^注は、シリアル通信を使用することでデバイスや統合開発環境 (IDE) に依存せずに開発を行うことができます。

注 QE for Capacitive Touch は、静電容量式タッチセンサを使用した組み込みシステム開発に必要なタッチ インタフェースの初期設定や感度調整に対応した開発支援ツールです。

動作確認デバイス

RL78/G22

静電容量センサユニット (CTSU) をサポートする RL78 ファミリ

目次

1.	概要	4
2.	動作環境	4
2.1	QE for Capacitive Touch の機能	5
3.	開発環境構築	6
3.1	開発ツールのインストール	6
3.1.1	統合開発環境 CS+のインストール手順	6
3.1.2	2 スタンドアロン版スマート・コンフィグレータのインストール手順	6
3.1.3	。 スタンドアロン版 QE for Capacitive Touch のインストール手順	6
3.2	ハードウェア設定	7
4.	アプリケーション開発手順	8
5.	アプリケーション例	. 10
5.1	アプリケーション例の概要	. 10
5.2	使用端子一覧	. 11
6.	新規プロジェクトの作成	. 12
7.	スマート・コンフィグレータの設定	. 13
7.1	スマート・コンフィグレータの起動	. 13
7.2	クロックとシステムの設定	. 14
7.3	SIS (Software Integration System) モジュールのダウンロード	. 15
7.4	コンポーネント追加	. 17
7.5	スマート・コンフィグレータによるコンポーネント設定の変更	. 19
7.5.1	CTSU コンポーネント設定	. 19
7.5.2	と Touch コンポーネント設定	. 21
7.5.3) UART 通信コンポーネント設定	. 22
7.5.4	・ LVD コンポーネント設定	. 24
7.5.5	う PORT コンポーネント設定	. 25
7.5.6	ぅ ボードサポートパッケージ	. 26
7.6	未使用端子の設定	. 27
7.7	コード生成	. 28
8.	QE for Capacitive Touch の設定	. 29
8.1	QE for Capacitive Touch の起動	. 29
8.2	プロジェクトの準備	. 30
8.3	タッチインタフェースの準備	. 32
8.4	調整	. 42
8.5	実装と動作確認	. 48
8.5.1	モニタリング	. 48
8.6	フローチャート(ソフトウェアタイマ)	. 55
9.	応用例	. 56



RL78 ファミリ FPB ボードでスタンドアロン版 QE を用いたタッチアプリケーショ	ョン開発
- 9.1 ハードウェアタイマでのタッチ計測	
9.1.1 スマード・コンフィッレーダの設定 (ハードウェアダイマ) 9.1.2 フローチャート (ハードウェアタイマ)	
9.1.3 サンフルコード (ハードウェアタイマ)	60
10. 参考ドキュメント	62
改訂記録	63



1. 概要

本アプリケーションノートでは、RL78 ファミリを使用した静電容量タッチ機能をシステムに組み込む、 以下の手順を説明します。

- RL78/G22 FPB を使用したスタンドアロン版スマート・コンフィグレータによるプロジェクト作成
- スタンドアロン版 QE for Capacitive Touch によるタッチインタフェース作成とチューニング及びモニタリング

本アプリケーションノートは、RL78/G22 FPB を使用して説明をしていますが、他の RL78 ファミリの静 電容量タッチ IP 搭載のデバイスにも活用することができます。

2. 動作環境

本アプリケーションノートの開発環境を表 2-1、表 2-2 に示します。

付属のサンプルコードは、表 2-1 に示すバージョンの開発環境で開発されています。また、本アプリケー ションノートは () 内に示すバージョンでの開発もサポートしています。

本アプリケーションノートでは、スタンドアロン版 QE にて生成したプログラムを CS+で RL78/G22 に書 き込み、書き込んだプログラムを RL78/G22 で動作させます。

項目	内容	バージョン
統合開発環境 (IDE)	CS+ for CC	V8.13.00
		(V8.09.00 以降)
コンパイラ	CC-RL	V1.15.00
		(V1.12.00 以降)
静電容量式タッチセンサ対応	スタンドアロン版 QE for Capacitive Touch	V4.1.0
開発支援ツール		
スマート・コンフィグレータ	RL78 スマート・コンフィグレータ	V1.12.00
		(V1.5.00 以降)
SIS (Software Integration	Capacitive Sensing Unit driver. (r_ctsu)	V2.10
System) モジュール	 Touch middleware. (rm_touch) 	(V2.10 以降)

表 2-1 開発環境 (ソフトウェア)

注意 タッチセンサのチューニング時に、CC-RL 無償評価版の V1.12.00 以降のバージョンを使用してコン パイルする場合は、コンパイラの最適化レベルを"デバッグ優先 (-onothing)"に設定してビルドし てください。

表 2-2 開発環境 (ハードウェア)

項目	内容
使用マイコン	RL78/G22 (R7F102GGE2DFB)
ターゲットボード	RL78/G22 Fast Prototyping Board (RTK7RLG220C00000BJ)

本アプリケーションノートのサンプルコードは、下記の条件で動作を確認しています。

表 2-3 動作確認条

項目	内容
動作電圧	5.0V
	LVD0 検出電圧:リセット・モード
	立ち上がり時 TYP.2.67V (2.59 V~2.75 V)
	立ち下がり時 TYP.2.62V (2.54 V~2.70 V)
動作周波数	高速オンチップ・オシレータ・クロック (fi⊦)∶32 MHz



2.1 QE for Capacitive Touch の機能

QE for Capacitive Touch は、静電容量式タッチセンサを使用した組み込みシステム開発に必要なタッチインタフェースの初期設定や感度調整に対応した開発支援ツールです。

QE for Capacitive Touch の主な機能は次のとおりです。

- タッチインタフェース構成の作成機能
 ボタン等のタッチインタフェースの配置とタッチセンサ(電極)の割り当てを視覚的に設定できます。
- チューニング機能
 タッチインタフェースのオフセットや感度を自動的にチューニングできます。
- モニタリングとパラメータ調整機能
 タッチインタフェースの動作をモニタリングでき、さらにパラメータの細かな調整を行うことができます。

Interface gene function	ration	Tuning function	Monitoring function	
Deser Configuration of Tour's Harrison		K) Automatic Taring Processing X	Stand Oracl X	N 00008***
Neved Kurt (N MA ImpCodipance	impe#/#z ctir	5/8: QE will now measure touch sensitivity for (Button01, TS23 @ config01).	Ver Type: Buttorcett, TS Proce TS23	
woor -	1 mm	In this step please use normal touch pressure on the sensor for once. Press any key on the PC keyboard to accept	Measurement Value: 11598 Baseline: 11481 Threshold: 2220 Stuch DIV/OFF difference: 117	
NEW NOV	Counter Tar	the sensitivity measurement.	Start Data Collection	
	Set Capacitant method in w	Button01, TS23 @ config01: 15419		
	Barra .			
	SMM PRODUCE			
	(Advised)	Canon	15449	
	store			
	Ryped			
	AT (1) of social		14444	
	Terrar			
	David Pe	THE THE PARTY AND A DECIDENCE OF A D		
	TC Fe	AANADA	13442	
	Opedanta Senior	Office Office In		
	La sel tras	B. Company P. Land C. Company and C.		
	000000	ATTING ON TO A T		
ne	Rence Into (P	" Quille Martin and Con Commander Martin	12440	1
Sing factor Sing Reviews Vite One Angred So	CONCEPTION OF THE OWNER	and the second and the second and the second		
		Credit in the state of the stat		

図 2-1 QE for Capacitive Touch の主な機能



3. 開発環境構築

本章では、開発環境のインストール手順およびハードウェアの設定について説明します。

3.1 開発ツールのインストール

本アプリケーション例では、以下のツールを使用します。

- CS+
- スタンドアロン版スマート・コンフィグレータ
- スタンドアロン版 QE for Capacitive Touch

すでに全てのツールをインストール済みの場合は、3.1節の手順は不要です。

- 3.1.1 統合開発環境 CS+のインストール手順
 - 下記リンクから、最新の "統合開発環境 CS+ for CC" のインストーラをダウンロードします。
 統合開発環境 CS+ | Renesas ルネサス
 - 2. ダウンロードした zip ファイルを解凍し、インストーラファイルを実行します。
 - 3. 「CS+のセットアップを開始する」をクリックします。
 - 4. "RL78 ファミリ用ツール"にチェックが入っていることを確認します。
 - 5. インストールが完了後、完了ボタンを押します。
- 3.1.2 スタンドアロン版スマート・コンフィグレータのインストール手順
 - 1. 下記リンクから、 最新の"RL78 スマート・コンフィグレータ"のインストーラをダウンロードします。
 RL78 スマート・コンフィグレータ | Renesas ルネサス
 - 2. ダウンロードした EXE ファイルを実行し、インストーラを起動します。
 - 3. インストーラを起動後、画面の手順に従ってインストールします。
- 3.1.3 スタンドアロン版 QE for Capacitive Touch のインストール手順
 - 下記リンクから、最新の "静電容量式タッチセンサ対応開発ツール QE for Capacitive Touch" のイン ストーラをダウンロードします。
 静電容量式タッチセンサ対応開発支援ツール QE for Capacitive Touch | Renesas ルネサス
 - 2. ダウンロードした zip ファイルには、プラグイン版 QE とスタンドアロン版 QE が同梱されていま す。ダウンロードした zip ファイルを展開し、スタンドアロン版 QE をインストールします。
- 注意 Windows のパス名の文字数制限 (260 文字) を超えないようにドライブのルートに近い場所に展開してください。

展開先の例: C:¥Renesas フォルダ以下

このとき、移動先として Windows フォルダや Program Files フォルダ、名前にスペースが含まれるフォルダなどは避けてください。



3.2 ハードウェア設定

ハードウェアの設定およびボード接続について説明します。表 3-1 に本アプリケーションでのターゲットボードにおけるジャンパの設定を示します。ボードへの電源供給については USB を使用します。図 3-1 のように PC とターゲットボードを USB ケーブルで接続します。

回路接続の変更内容は、ターゲットボードによって異なります。詳細は使用する FPB のユーザーズマ ニュアルの「QE for Capacitive Touch 使用時の注意事項」をご確認ください。

位置	回路グループ	ジャンパ設定	用途
J16 ^注	QE シリアル接続切り替え	オープン	QE のシリアル接続機能を実行する場合
	シャンハ	ショート	COM Port デバック回路を有効にする場合
J17	電源選択ヘッダ	1-2 ショート	5V 電源を選択

表 3-1 ボードのジャンパ設定

注 本開発手順では、チューニング時およびモニタリング時に J16 の設定を切り替えます。 詳細は、8.4 調整の手順3、手順4、手順5 をご確認ください。



図 3-1 PC とターゲットボードの接続

4. アプリケーション開発手順

アプリケーション開発の手順を説明します。

開発の流れは QE for Capacitive Touch のワークフローに従います。



図 4-1 アプリケーション開発手順

ワークフローの各項目を表 4-1 に示します。本表の章番号は、関連章にリンクされています。章番号をク リックして使い方を確認してください。プロジェクトの作成、プログラムの実装、プロジェクトのビルド、 デバッグの開始は、IDE やスマート・コンフィグレータを使います。



項目			章番号
プロジェクトの準備	プロジェクトの作成	IDE によるプロジェクトの作成	6
		スマート・コンフィグレータの設定	7
		クロックとシステム	7.2
		CTSU 用ドライバ	7.5.1
		タッチ用ミドルウェア	7.5.2
		シリアルインタフェース (UART 通信)	7.5.3
		電源検出回路 (LVD)	7.5.4
		ポート機能 (PORT)	7.5.5
		ボードサポートパッケージ	7.5.6
		未使用端子の設定	7.6
	プロジェクトフォルダ	の選択	8.2
	マイコンの選択		
タッチインタフェースの	構成の選択		8.3
準備	調整用ファイルの出力	1	
	プログラムの仮実装		
	プロジェクトのビルド		
調整	プログラムの実行		8.4
	調整の開始		
	調整結果の出力		
実装と動作確認	プログラムの実装		8.5
	デバッグの開始		
	シリアル接続		
	モニタリングの開始		

表 4-1	QE for Ca	apacitive Touch	nを用いた開発の	ワークフロー
-------	-----------	-----------------	----------	--------



5. アプリケーション例

5.1 アプリケーション例の概要

本アプリケーションノートでは、2つのボタンおよび1つのスライダを使用するアプリケーションを例に 説明します。また、シリアル通信を介してタッチ性能のチューニングとモニタリングを行います。

6章以降で、2つのボタンおよび1つのスライダを使用するアプリケーションを作成し、ボタンまたはス ライダをタッチした場合の検出状況をモニタリングする方法を示します。

備考 タッチアプリケーションのタッチ性能のモニタリングは、OCD (On-Chip Debugging) エミュレータ
 を介した通信によっても確認できます。ただし、RL78 ファミリの場合、モニタリングパフォーマン
 スは、RL78 ファミリのオンチップ・デバッグ機能によって制限されます。
 シリアル通信を介してタッチ性能のモニタリングを行うことで、スムーズなモニタリングが可能に
 なります。また、タッチセンサのチューニングもシリアル通信を介すことができます。



図 5-1 アプリケーション例

本アプリケーションノートには、表 5-1 に示した2つのサンプルコードが付属しています。両サンプル コードは、開発手順は共通ですが、スマート・コンフィグレータの設定と qe_touch_sample.c ファイルへ追 加するコードが一部異なります。以降の6章から8章では、ソフトウェアタイマを用いたタッチアプリケー ションを例にして、開発手順を説明しています。ハードウェアタイマと LED 制御の実装方法については 「9. 応用例」を参照してください。

表 5-1	付属のサンプルコー	ドの概要
-------	-----------	------

ファイル名	タッチ計測周期を生成するタイマ	LED 制御
Capacitive_Touch_Project_Example	ソフトウェアタイマ	無し
Capacitive_Touch_Project_HardwareTimer_Example	ハードウェアタイマ	有り

5.2 使用端子一覧

本アプリケーション例の使用端子を、表 5-2 に示します。

UART 通信およびタッチセンサは、使用するターゲットボードの仕様にあわせて設定する必要があります。

項目	端子	用途
UART 通信	RxD0/P11	チューニング、モニタリングを
	TxD0/P12	行う
タッチセンサ①	TS24/P26	ボタンでのタッチを検出する
		(TS_B1)
タッチセンサ②	TS23/P25	ボタンでのタッチを検出する
		(TS_B2)
タッチスライダ	TS20/P22	スライダ上の、指が左右に移動
	TS21/P23	した際の位置を検出する
	TS22/P24	(TS_S)

表 5-2 本アプリケーション例の使用端子一覧

本アプリケーション例で使用するタッチセンサの配置を図 5-2 に示します。



図 5-2 タッチセンサの配置

6. 新規プロジェクトの作成

CS+を起動し、プロジェクトを作成します。

"プロジェクト作成"ダイアログにて、本アプリケーションでは以下を選択します。

- マイクロコントローラ(T)
- 使用するマイクロコントローラ(<u>M</u>)
- プロジェクトの種類(K)
- プロジェクト名(N)
- 作成場所(L)

: RL78 : R7F102GGExFB (48pin) : アプリケーション (CC-RL) : (任意のプロジェクト名) : (任意の作成場所)

プロジェクト作成		×
マイクロコントローラ(T):	RL78	\sim
使用するマイクロコントローラ(<u>M</u>)		
🏭 (マイクロコントローラを検索で	(************************************	
R7F 102GBE×NP(3) R7F 102GCE×LA(3) R7F 102GCE×NP(4) R7F 102GGE×NP(4) R7F 102GGE×FP(44 R7F 102GGE×FP(44 R7F 102GGE×NP(4) R7F 102GE×NP(4) R7F 102GE×N	2pin) 5pin) 5pin) 1pin) 1pin) 8pin) 8pin) B)	×
プロジェクトの種類(K):	アプリケーション(CC-RL)	~
プロジェクト名(N):	Capacitive_Touch_Project_Example	
作成場所(L):	C:¥CS+_Workspace	✓ 参照(R)
\implies	🗹 プロジェクト名のフォルダを作成する(A)	
C:¥CS+_Workspace¥Capacitive	e_Touch_Project_Example¥Capacitive_Touch_Project	t_Example.mtpj
□ 既存のプロジェクトのファイルね	構成を流用する(S)	
流用元のプロジェクト(P):	(流用元のプロジェクト・ファイルを入力してください)	~ 参照(W)
□ プロジェクト・フォルダ以下の株	構成ファイルをコピーして流用する(O)	
マルチコア用オブション付きダイブ	PDグへ(1) 作成(C) キャンセル	ヘルプ(H)

図 6-1 プロジェクトの作成

7. スマート・コンフィグレータの設定

スマート・コンフィグレータの設定手順を説明します。本アプリケーション例で必要な設定は、以下のと おりです。

- クロックとシステム
- CTSU 用ドライバ
- タッチ用ミドルウェア
- シリアルインタフェース (UART 通信)
- 電圧検出回路 (LVD)
- ポート機能 (PORT)
- 7.1 スマート・コンフィグレータの起動

CS+の"プロジェクト・ツリー"で、"スマート・コンフィグレータ"をダブルクリックし、起動します。



図 7-1 スマート・コンフィグレータを起動

上記方法でスマート・コンフィグレータが起動しない場合は、以下の2点を確認してください。

- スマート・コンフィグレータのプロパティで"ファイルパス"が正しく設定されているか
- メニューの [ツール] [プラグインの管理]から、"RL78 用スマート・コンフィグレータ通信プラグイン" がチェックされているか

לםטֿגלולישי א א	1 기미パティ 🥑 mainc 🧃 qe_touch_sample.c		+ ×
	 □ スマート・コンパグレータのプロパティ マ スマート・コンパグレーク協定 同じ時代スマート・コンパグレークGose77イルパス >> 製品情報 パークョン 	OVProgram Files (x88)WRenesas ElectronicsVSmartConfiguratorVRL78VeclipseVSmartConfiguratorexe	- +

図 7-2 スマート・コンフィグレータのファイルパス

プラグインの管理	×
CS+起動時に読み込むプラグインにチェックしてください。 この設定は次回起動時に有効となります。 ※CS+の動作に必須のプラグインはグレー表示となっており、 なるマイクロコントローラ用プラグインのチェックは、外さないこと 基本機能」追加機能	チェックを外すことはできません。また、基本機能タブにおいて、開発対象と とを推奨します。
 モジュール名 IronPythonコンソール・プラグイン Quick and Effective tool solution - QE 留 RL78用スマート・コンフィグレータ通信プラグイン アップデート・マネージャ・プラグイン 	説明 IronPythonのコマンドとCS+拡張機能が使用できるコンソールです。 アプリケーション開発に便利なツールをセットにしたプラグインです。 ドライバの生成とミドルウェアのインポートを行うRL78用スマート・コンフィ CS+ アップデート・マネージャと連携するプラグインです。

図 7-3 プラグインの管理



7.2 クロックとシステムの設定 クロックとシステムの設定手順を説明します。

 スマート・コンフィグレータを起動後、Smart Configurator view の下部にある[クロック]タブを選択し、 クロックを設定します。EVDD がある MCU の場合は、動作モードに合わせて"EVDD 設定"を行ってくだ さい。

カロック設定		5
		コードの生成 レポートの生成
動作モード	黒道メインドモード2.7(V)~5.5(V)	
✓ 高速力: 周边数: 所OCO開始 (STOPE→R 発展器を起想)	7-17/0-9 32 (MHz) 2: ian 	IHP 32 (MHz) MAIN 32 (MHz) S2 (MHz) (MHz) 32 (MHz) (MHz) S2000 (HHz) (MHz)
	7.451	
用波数:	4 (MHz)	(ter sc)
X1発援0	公用語 2 (1) (1) (1) (1) (1) (1) (1) (1	
動作モード:	X10.6 ·	- (MHz)
周波数:	5 (MHz)	
光振安定時間	2*18/fx + 52428.8(µs)	

図 7-4 クロックの設定

EVDD設定: 2.7 V ≤ EVDD0 ≤ 5.5 V ▼	動作モード:	高速メイン・モード2.7(V)~5.5(V)	•
	EVDD設定:	2.7 V ≤ EVDD0 ≤ 5.5 V	•

図 7-5 EVDD 設定

2. [システム]タブを選択し、デバッグ環境を設定します。

システム設定			▶ 📟 コードの生成 レポートの生成
3			
オンチップ・デバッグ設定			
オンチップ・デバッグ動作設定 〇 使用しない	○ エミュレータを使う	● com#-ト	
- エミュレータ設定 ○ E2	E2 Lite	•	
疑似RRM/DMM機能設定 ○使用しない	● 使用する		
Start/Stop関数機能設定 ●使用しない	○使用する		
- 通過ボイント機能設定 ◎ 使用しない	○ 使用する		
セキュリティID設定 し セキュリティIDを設定する セキュリティID	チェックを外す		
セキュリティID認証失敗時の設定 フラッシュ・メモリのデータを消去しな フラッシュ・メモリのデータを消去する 	:U ;		

図 7-6 デバッグの設定



7.3 SIS (Software Integration System) モジュールのダウンロード

タッチアプリケーションの実装に必要な"CTSU用ドライバ"と"タッチ用ミドルウェア"の2つのSISモジュールのダウンロード方法を説明します。すでにダウンロード済みの場合は、本項は不要です。

1. [コンポーネント]タブを選択し、゛ アイコンをクリックします。

ソフトウェアコンボーネント設定 ご コンボーネント 油 山 小 日 田 設定 ① 「マルタスカ 「マルタスカ」 「マルタスカ」
□ 2 xm - ネント 油 凶 № 日 田 酸定 00 ■ 2 km - 2 xm - 2
 ✓ ▲ x>-Fy*y/ ✓ ▲ x>-Fy*y/ ✓ t bsp

図 7-7 ソフトウェアコンポーネント設定画面

2. "コンポーネントの追加"ダイアログの下側にある"RL78 Software Integration System モジュールをダウン ロードする"をクリックします。

עב 🛐	ポーネントの追加				\times
ソフト・	フェアコンポーネントの選択				
使用了	「能なコンポーネントの一覧から選択してく	(ださい)			
12/11		1200			
カテブリ	全7				\sim
100	<u></u>				~
100 HC	±(
フィルろ					
לעב		Short Name	タイプ	バージ	^
# A/	Dコンバータ		コード生成	1.6.0	
# Bo	ard Support Packages v1.80	r_bsp	RL78 Software	1.80	
🖶 Ca	pacitive Sensing Unit driver.	r_ctsu	RL78 Software	2.10	
🖶 IIC	通信 (スレープ・モード)		コード生成	1.6.0	
IIC 🖶	通信 (マスタ・モード)		コード生成	1.7.0	
# PV	VM出力		コード生成	1.8.0	
🚓 SN	IOOZEモード・シーケンサ		グラフィカル・コン	1.3.2	
₩ SP	I(CSI)通信		コード生成	1.6.0	
the To	uch middleware.	rm_touch	RL78 Software	2.10	
₩ UA	ART通信		コード生成	1.8.0	
₩ 1/	ベントリンクコントローラ		コード生成	1.3.1	
■ イン	ソターバル・タイマ		コード生成	1.6.0	
申ウ:	tッチドッグ・タイマ		コード生成	1.6.0	\sim
☑最	新バージョンのみ表示				
説明					
アナロ	グ-デジタル(A/D)変換回路は、アナログ	入力をデジタル信号に変	変換する機能です。		\sim
					\sim
<u>RL78</u>	Software Integration Systemモジュール				
基本語	<u>čæ</u>				
?	< 戻る(B)	次へ(N) >	終了(F)	キャンセル	,

図 7-8 コンポーネントの追加ダイアログ



表示されたダイアログで以下を選択し、ダウンロードをクリックします。
 — RL78 ファミリ CTSU モジュール Software Integration System
 — RL78 ファミリ TOUCH モジュール Software Integration System

RL78 Software Integration Systemモジュールの	のダウンロード				
ダウンロードするRL78 Software Integration Systemモジュ	ールを選択してください	1 ₀			
91hu	ドキュメントNo.	リビジョン	発行日	^	すべて選択
RL78 Family Renesas ZMOD4410, ZMOD445	R01AN6197EJ01	Rev.1.40	2024-10-23		選択をすべて解除
▶ ☑ RL78ファミリ TOUCHモジュール Software Integra	R11AN0485JJ02	Rev.2.10	2024-10-15		
▶ ☑ RL78ファミリ CTSUモジュール Software Integratio	R11AN0484JJ02	Rev.2.10	2024-10-15		
RL78ファミリ Renesas Flash Driver RL78 Type 0	R20AN0656JJ01	Rev.1.20	2024-05-20		
RL78ファミリ Renesas Flash Driver RL78 Type 0	R20AN0655JJ01	Rev.1.20	2024-05-20		
RL78ファミリ Renesas Flash Driver RL78 Type 0	R20AN0654JJ01	Rev.1.20	2024-05-20		
RL78ファミリ Renesas Flash Driver RL78 Type 0	R20AN0653JJ01	Rev.1.20	2024-05-20		
RL78ファミリ Serial NOR Flash Memory 制御モ	R01AN7243JJ01	Rev.1.00	2024-03-22		
□ RL78ファミリ ボードサポートパッケージモジュール Soft	R01AN5522JJ01	Rev.1.62	2023-11-30	~	
<			>		
モジュール・フォルダー・パス:					
					参照

図 7-9 SIS モジュールのダウンロード

注意 上記ダウンロード画面に、TOUCH モジュールおよび CTSU モジュールが表示されない場合は、別の 手順でダウンロードする必要があります。 Renesas Web から各モジュールのダウンロードを行い、手順に沿って SIS モジュールのダウンロー ドフォルダにファイルを追加します。

各モジュールのウェブページおよび適用方法については以下を参照してください。

• CTSU モジュールと TOUCH モジュールのダウンロード

RL78 ファミリ CTSU モジュール Software Integration System RL78 Family CTSU Module Software Integration System Rev.2.10 - Sample Code | Renesas ルネサス

RL78 ファミリ TOUCH モジュール Software Integration System RL78 Family TOUCH Module Software Integration System Rev.2.10 - Sample Code | Renesas ルネサス

• 各モジュールの適用方法

<u>Renesas Web 等からダウンロードした Software Integration System モジュールの適用方法 | Renesas</u> <u>Customer Hub</u>



7.4 コンポーネント追加

1. スマート・コンフィグレータで以下のコンポーネントを選択してください。



図 7-10 ソフトウェアコンポーネントの選択

注 RL78/G16 の場合は電圧検出機能の設定方法が異なりますので、コンポーネントで本機能を追加する 必要はありません。 RL78/G16 での設定方法は、7.5.4 節をご確認ください。



2. 選択したコンポーネントに対してリソースを設定します。本アプリケーション例では以下の設定で使用 します。

ראלי אינג מיני אינג אינג אינג אינג אינג אינג אינג א			
「「「」、「」、「」、「」、「」、「」、「」、「」、「」、「」、「」、「」、「」			
選択したコンポーネント	のコンフィグレーションを追		
<u>Л</u> ЦС <i>ж</i> 9			
UART通信			
コンフィグレーション名:	Config_UART0		
顺力 作:	送信/受信		\sim
リソース:	UARTO		\sim
+* L			
コンフィグレーション名:	Config PORT		
リソース:	PORT		~
電圧検出回路	Config IVD0		
コンフィッション名・			
	LVDO		· ·
?	< 戻る(B) 次へ(N) >	終了(F)	キャンセル

図 7-11 コンポーネントのリソース設定

以下に示すようにコンポーネントが追加されます。

*Capacitive Touch Project Example.scf	a ×		
いつトウェアコンポークントシッテ			
シントウエアコンホーホント設定			
コンポーネント 🚵 🚵 🗠 🖻 🖽	設定		
 編 認	プロパティ ~ @ Configurations	値	
× 🕞 74-FPUJ	# Parameter check	Use system default	
 シレントレント シレントレント シレントレント シレントレント シレントレント シレント 	# Data transfer of INTCTSUWR and INTCTSU	Interrupt handler	
e r bsp	# DTC setting	Setting in r_ctsu	
> シートライバ	# Select auto judgement	Disable	
~ 🗁 電源管理とリセット機能	# Auto-judgment function in Snooze mode	Disable	
Config_LVD0	# Data storage address setting for CTSURD		
~ ├ 入出カポート	# Data storage address setting for CTSUWR		
Config_PORT	# Interrupt level for INTCTSUWR	Level 2	
✓ → 通信	# Interrupt level for INTCTSURD	Level 2	
Config_UART0	# Interrupt level for INTCTSUFN	Level 2	
、 シー ミドルウェア	# Output port number for external trigger	PORT14	
、 ら ジェネリック			
💣 r_ctsu			
💣 rm_touch			
柳栗 ボード クロック シフテム コンボーネント	業子 割け込み		
「彼安」ホート「シロッシーシスノム「コノルーネント」			

図 7-12 ソフトウェアコンポーネント設定(コンポーネント追加後)



7.5 スマート・コンフィグレータによるコンポーネント設定の変更

追加した各コンポーネントについて設定を行います。

7.5.1 CTSU コンポーネント設定

"r_ctsu"モジュールをクリックします。TSCAP 端子とアプリケーションで使用する5つのTS 端子を有効 にします。TS 端子とタッチセンサの割り当ては、使用するターゲットボードのユーザーズマニュアルをご 確認ください。

$^{\oplus}$ *Capacitive_Touch_Project_Example.scfg \times		
ソフトウェアコンポーネント設定		⑤ コードの生成 レポートの生成
בעל-אָרָר 🚵 🖬 🖓 🗉 🖽	設定	Œ
56 kš	プロパティ	値
フィルタ入力	✓ ⊕ Configurations	
x @ 78-17W7	# Parameter check	Use system default
	# Data transfer of INTCTSUWR and INTCTSURD	Interrupt handler
2 r ben	# DTC setting	Setting in r ctsu
N CA FEAR	# Select auto judgement	Disable
マ 四、電道管理と日から場的	# Auto-judgment function in Snooze mode using SMS	Disable
Config IVD0	# Data storage address setting for CTSURD	0xFF500
	# Data storage address setting for CTSUWR	0xFF800
Config PORT	# Interrupt level for INTCTSUWR	Level 2
v DA 通信	# Interrupt level for INTCTSURD	Level 2
Config LIARTO	# Interrupt level for INTCTSUFN	Level 2
 Coning_OARTO Coning_OARTO Coning_OARTO 	# Output port number for external trigger	PORT14
× (P) ジェネリック	# Bit number for exremal trigger output	BITO
r ctsu	# Interrupt port number for external trigger	INTP1
a m touch	✓ ^{(■} リソース	
m_toden	v 🖸 CTSU	
	► TSCAP端子 TSCAP端子	📝 使用する
	~ TS00端子	◎ 使用する
	~ TS01端子	🗐 使用する
	~ TS02端子	□ 使用する
	~ TS03端子	🔄 使用する
	~ TS04端子	🔄 使用する
	~ TS05端子	回 使用する
	~ TS06端子	回 使用する
	~ TS07端子	回 使用する
	~ TS08端子	🔄 使用する
	~ TS09诸子	🔄 使用する
	~ TS10诸子	🔄 使用する
	~ TS11端子	🔄 使用する
	~ TS12端子	🔄 使用する
	~ TS13端子	🗐 使用する
	~ TS14端子	🔄 使用する
	➡ TS15端子	🔄 使用する
	~ TS16端子	🗐 使用する
	~ TS17端子	🔲 使用する
	~ TS18端子	📃 使用する
	~ TS19ن着子	🔄 使用する
	~ TS20端子	☑ 使用する
	► TS21端子 TS20端子	☑ 使用する
	~ TS22端子	 使用する
	► TS23端子 TS24端子	☑ 使用する
	~ TS24端子	 使用する
	~ TS25端子	回 使用する
	~ TS26端子	■ 使用する
	~ TS27端子	🔄 使用する
	~ TS28端子	🔄 使用する
	4	

図 7-13 アプリケーションで使用する TS 端子の有効化



また、アプリケーションで使用しない TS 端子については、Low レベル出力に設定することを推奨します。CTSU2 では、アプリケーションで使用しない TS 端子を有効にした場合、非計測端子として、Low レベル出力の設定になります。

そのため、付属の2種類のサンプルコードでは、兼用機能 (UART0) を割り当てて使用する TS12 / TS13 端子を除き、アプリケーションで使用しない TS 端子についてもすべて有効にしています。

注意回路を作成する際は、端子処理などを適切に行い、電気的特性を満たすようにしてください。



図 7-14 アプリケーションで使用しない TS 端子の有効化



7.5.2 Touch コンポーネント設定

"rm_touch"モジュールをクリックし、以下を設定します。

- Support QE monitor using UART Enable
- Support QE tuning using UART Enable
- UART channel UART0

設定する UART チャネルは、使用するターゲットボードによって異なります。詳細は使用する FPB の回路図をご確認ください。

値 Use system default Enable Enable UARTO TypeA : Counter of exceed threshold is ho	*
	Use system default Enable Inable UART0 TypeA : Counter of exceed threshold is ho

図 7-15 rm_touch の設定



7.5.3 UART 通信コンポーネント設定

タッチセンサのチューニング、モニタリングで使用する UART の設定手順を説明します。

設定する UART チャネルおよびポートは、使用するターゲットボードによって異なります。追加した"UART 通信"モジュールをクリックし、目標の転送レート (ボーレート) に合わせて "送信" および "受信" タブにて動作クロックと転送レートを選択します。

本アプリケーションノートでは、以下のように設定します。

			□-ドの生成 レポー
시(* 전 1월	⊖ 🗷 設定		
*	送信 🔚		
入力	UART0クロック設定		
スタートアップ	動作クロック	скоо 🗸 🥌	
ジェネリック ♪ r hsp	クロック・ソース	fCI K/2^3 、 v / 读数: 4000 kHz)	
ドライバ			
≥ 電源管理とリセット機能			
Config_LVD0		● 建机构区 L-1.	
Config_PORT		OPFWL	
▶ 通信		0.2.7	
Config_UARTO	● LSB	MSB	
ミトルフェア 参 ジェネリック	パリティ設定		
💣 r_ctsu	● パリティ・ビットなし ○ 0パリティ	○ 奇数パリティ ○ 偶数パリティ	
rm_touch	ストップ・ビット長設定		
	 1ยังห 	O2Ľット	
	送信データ・レベル設定		
	● 非反転(通常)	○反転	
	転送レート設定		
	転送レート設定	153600 v (bps) (bps)	
	割り込み設定		
	送信完了割り込み設定(INTSTO)	レベル3(低優先順位) ~	
	コールバック機能設定		
	□ 送信完了		
ド クロック システム コンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定	副 952.87		
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定	新り込み 		で〕 コードの生成 レポーI
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × ウェアコンボーネント設定 たント ごご ピ ペ			コードの主成 レポー
ド クロック システム <u>コンボーネント</u> 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 キント <u></u>	割り込み □ ● ● 設定 送信 受信		で) コードの主成 レポー1
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 キント <u></u> <u></u> 企 代。 、 入力 スタートアップ	割り込み 日 世 設定 送信 受信 UART02Dック設定		で) コードの生成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 キント <u></u> 企 代。 1 入力 スタートアップ 会 ジェネリック	割り込み ・ 世 設定 送信 受信 UART02Dック設定 動作クロック	СК00 ~	で) コードの主成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 キント <u><u></u> 12 (2) ト カカ スタートアップ ディルタック ディレSpp ドラムア</u>	割り込み ・ 世 設定 送信 受信 UART02Dック設定 動作クロック クロック・ソース	CK00 CK00 fCLK/2^3 K250: 4000 kHz)	う コードの主成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × アエアコンボーネント設定 キント 油 凶 役 () キント 油 凶 役 () シカ スク スク アップ シゴネリック で, bsp ドライバ ● 電源管理とりセット機能	 割り込み ■ 設定 送信 受信 UART02Dック設定 動作クロック クロック・ソース データ・ビット長設定 	CK00 KCLK/2^3 CK00 kHz)	う コードの主成 レポー
ド クロック システム コンボーネント 端子 citive_Touch_Project_Example.scfg × クェアコンボーネント設定 キント シュ ユーネント スク スク スク アップ ・ 「Losp ドライバ ● 電源管理とりセット機能 ● Config_LVD0	 割り込み ■ 設定 送信 受信 UART02ロック設定 動作クロック クロック・ソース データ・ビット長設定 ○ 7ビット ● 8ビット 	СК00 КСЦК/2^3 ЭЕ'ун	で) コードの主成 レボー
ド クロック システム コンボーネント 端子 citive_Touch_Project_Example.scfg × ウェアコンボーネント設定 キント シュ ユートアップ シスタートアップ ・ ブム スタートアップ ・ ブムキリック ・ てのfig_LVD0 シ ムカボート ・ Config_DVD1	 割り込み 副り込み 法信 受信 UART02D975股定 助作クロック クロック・ソース アーク・ビット長設定 7ビット ● 8ビット デーク転送方向設定 	СК00 КСЦК/2^3 ЭЕ'ун-	う-ドの生成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × DT アコンボーネント設定 Ryh	 割り込み ■ 設定 送信 受信 UART0クロック設定 動作クロック クロック・ソース データ・ビット長設定 ○ 7ビット ● 8ビット データ転送方向設定 ● LSB ビット 	СК00 <u>тСLK/2^3</u> 9УУ- МSB	で」 コードの生成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 キント 2010日 キント 2010日 ネカカ スタートアップ シブェネリック マートアップ マートアップ マートアップ シブ・ マートアップ マート マートアップ マートアップ マートアップ マートアップ マートアップ マートアップ マートアップ マートアップ マート マート マートアップ マート マート マートアップ マート マート マート マート マート マート マート マート	 割り込み 割り込み 送信 受信 送信 受信 UART0クロック設定 動作クロック クロック・ソース データ・ビット長設定 ○Tビット ⑧ 8ビット データ転送方向設定 ⑥ LS8 パリティ設定 ○ロリュット どいたちょ ○ロリュット どいたちょ ○ロリュット どいたちょ ○ロリュット どいたちょ ○ロリュット ジャルちょ ○ロリュット ジャルちょ ○ロリュット 	СК00 • <td>で」 コードの生成 レポー1</td>	で」 コードの生成 レポー1
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 とシト スクートアップ スカ スタートアップ ジスオリック ごのり スカ スタートアップ ジスオリック ごのしての前夏_UVD® ごのの前夏_UART0 ミンコンボート ごのの前夏_UART0	 割り込み ⇒ ⇒<	CK00 	で」 コードの主成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × シ フェアコンボーネント設定 シアノアンボーネント設定 シント シン シント シン シン スク シン スクートアップ シン シブトアップ シン ジェネリック こちの ドンクレ シン マートアップ シン シブトアップ シン シスカ マン スクートアップ シン シント シン シン この シブレン マン シス倍 Config_UART0 シドレクエア シン シブドメリック マーちの ジェスシック マーちの	 割り込み ⇒ 数定 送信 受信 UART0クロック設定 動作クロック クロック・ソース データ・ビット長設定 ○ 7ビット ● 8ビット データ転送方向設定 ● LSB パリティ投定 ③ パリティイとットなし、○ 0パリティ ストップ・ビット長設定 1とり ストップ・ビット長設定 マバリティビットなし、○ 0パリティ ストップ・ビット長設定 	CK00 「」 「」	で」 コードの主成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 キント 逆 20 % キント 逆 20 % キント 逆 20 % キント 逆 20 % キント ご 20 % キント ジ 20 % キント ジ 20 % キント ジ 20 % キント ご 20 % キント ご 20 % キント ご 20 % キント ジ 20 % キント シント シント シント シント シント シント シント シント シント シ	 割り込み ● ● 設定 送信 受信 UART002097設定 助行クロック クロック・ソース アーク・ビット長設定 ○ 7ビット ● 8ビット データ転送方向設定 ● LSB パリティ設定 ○ パリティ・ビット表設定 1ビット展設定 ビット調査 ○ パリティビット表設定 1ビット展設定 ビット ビット ビット 	KK00 KCLK/2^3 ・ 9ビット ・ ・ MSB ・ 奇数パリティ	で] ⊐-ドの主成 レポー
ド クロック システム コンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 キント 油 は Pc 1 ネカ スク スクートアップ ジェネリック で、tops トライパ ● 満添管理とりセット機能 ● Config_LVD0 > 入出力ボート ● Config_PORT > ジェネリック * JConfig_UART0 ミドルウェア > ジェネリック * r_ctsu * r_mtouch	 割り込み 割り込み ③ 送信 受信 UART00D9/0股定 助作クロック クロック・ソース アータ・ビット長設定 フビット アーク転送方向設定 ⑥ 8ビット デーク転送方向設定 ⑥ (リフィビット長設定 (ワリティビットなし) ○ のパリティ ストップ・ビット長設定 1ビット マ(リティビットなし) ○ のパリティ ストップ・ビット長設定 1ビット マ(リティ・ビットなし) ○ のパリティ ストップ・ビット長設定 1ビット マ(リティ・ビット マ(リティ・ビット シーンパル設定 ● まら転(海索) 	CK00 fcLK/2^3 9ビット のSB 可数パリティ 反反転	で) コードの主成 レポー
ド クロック システム コンボーネント 端子 citive_Touch_Project_Example.scfg × リエアコンボーネント設定 キント 油 20 % シンカ スクートアップ ジエネリック で、cbsp トライパ ● Config_LVDD > 入面ボート ○ Config_LVDT ● YIR4JUPD ● T_CtSu ● r_ctSu ● r_ctSu ● r_m_touch	割り込み ● ① 設定 送信 受信 UART00Dック設定 助作クロック クロック・ソース アータ・ビット展設定 アンマル ● LSB パリティ設定 パリティシンシーム シークシャンシース ジークを送方向設定 ● LSB パリティ設定です 受信データトレベル設定 ・アシークテレベル設定 ・ デジートンパル設定 ・ デジートシーシット ・ デジート・シーシー ・ デジー・レベル設定 ・ デジー・レベル設定 ・ デジー・レベル設定 ・ デジー・レベル設定 ・ デジー・レベル設定	CK00 ・ FCLK/2^3 ・ 9ビット ・ 小SB ・ ・ 新数パリティ ・ 新数パリティ	で] ⊐-ドの主成 レポー
ド クロック システム コンボーネント 端子 citive_Touch_Project_Example.scfg × リアコンボーネント設定 キント シュースント キント シュースント オフ スクートアップ ジェネリック で、LSP マートアップ ジェネリック で、Config_LVD0 ン出カボート ご Config_LVART0 ドルクエア ジェネリック ・「このfig_UART0 ドルフェア ジェネリック ・「こたちu ・「m_touch	別リ込み 別リ込み 送信 受信 送信 受信 以ART0クロック対象定 動作クロック クロック・ソース データ・ビット長設定 〇アビット ③ 8ビット データ転送方向設定 ③ 101 ティビット長設定 ビット 須定です 受信データ・レベル設定 ● 非反転(通常) 転送レート設定 転送しート設定	CK00 「 fcLK/2^3 「 9ビット 」 の 所SB () 奇数パリティ 一 奇数パリティ () 偶数パリティ	で) □-ドの主成 レポー
 ド クロック システム ユンボーネント 端子 ボ クロック システム ユンボーネント 端子 ロアコンボーネント設定 キント 2012 (2015) キント 2012 (2015) キント シュ 2015) シュ 2015) ジェネリック ・ こops ・ こonfig_LVD0 シ 入出カボート ・ Config_LVAT0 ドルクェア ジェネリック ・ こonfig_LORT ラ 通信 * Config_UART0 ドルクェア ジェネリック ・ こcutau ・ 「 m_touch 	別以込み 別以込み 送信 受信 送信 受信 UART0クロック設定 動作クロック クロック・ソース データ・ビット長設定 〇 7ビット ⑧ 8ビット データ転送方向設定 ⑨ パリオ・ビット長設定 パリオ・ビット長設定 ビット固定です 受信アータ・レベル設定 ● 非反転通常) 転送レート設定	CK00 ・ FCLK/2^3 ・ ・ 9ビット ・ ・ ・ 9ビット ・ ・	「□ (□-ドの生成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × クェアコンボーネント設定 キント 2000 1000 1000 1000 1000 シスカ スタートアップ ジェネリック で、cbsp ドライバ 電子管理とりセット機能 で、Config_PORT ジェネリック ジェネリック で、ctsu で、Config_UART0 EFルウェア ジェネリック で、ctsu で、m_touch	割り込み 部定 送信 受信 送信 受信 UART02D97設定 動作クロック クロック・ソース データ・ビット長設定 〇アビット ● 103 「リフィ設定 ● パリティどット長設定 〇アビット ● パリティビット ● パリティビット ● ポリティ設定 受信アータ・レベル設定 ● 非反転(通常) 転送レート設定 転送レート設定 副比込み物学	CK00 ・ fcLK/2^3 ・ 9ビット の MSB ① 奇数パリティ ○ 反転 153600 ・ (誤差: 0.16%,許容最小::-4.49%,許容最大: : 4.42%)	で」 コードの生成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.sctg × クェアコンボーネント設定 シュースシートおりた マクトアップ シュスカリック マクトアップ シュスカリック マクトアップ シュートアップ シュスカリック マクトアップ シュートアップ シュートアップ シュートアップ マクトアップ マクトアップ シュートアップ シュートアップ シュートアップ マクトアップ マクトアップ シュートアップ シュートアップ マクトアップ マクトアップ シュートアップ シュートアップ マクト マクトアップ マクトアップ マクトアップ マクトアップ マクトアップ マク マクトアップ マクトアップ マクトアップ マクトアップ マクトアップ マクト マクトアック マクト マクトアップ マクトアック マクトアック マクト マクトアック マクトアック マクト マクトアック マク マクトアック マクトアック マク マクトアック マク マク マクトアック マクトマーク マク マク マク マク マク マク マク マク マク マク マク マク マク	割切込み ③ ③ ③ ③ ③ ③ ③ ③ ③ ③ ③ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	CK00 ・ 「CLK/2^3 ・ ● 9ビット ・ ● 9ビット ・ ● かあめパリティ ● 偶数パリティ ● 万転 ・ 153600 ・ () 既転 ・ () 既転 ・ () 既転 ・ () 既転 ・ () 近日の気管電量値() ・ ・ ・ <	で」 コードの主成 レボー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × フェアコンボーネント設定 ドンボーネント設定 ドンパ ションボラック ションボラック ジェネリック ジェネリック ジェネリック ジェネリック ションボート Config_LVTV機能 ごのfg_DORT ジェネリック ジェント ジェック ジェント	割り込み 割り込み こ 送信 受信 送信 受信 UART0クロック設定 動作クロック クロック・ソース データ・ビット長設定 〇アビット ③ 8ビット データ転送方向設定 ③ 1/リティ検定 ③ 1/リティ検定 ③ 1/リティ・ビット長設定 ビット展定です 要信データ・レバル設定 ● 非反転(通常) 転送レート設定 割り込み設定 >関リ込み設定 ごったり、決力設定(INTSR0) □ コーカリシカ会迎(INTSR0)	CK00 ・ 「CLK/2^3 ・ ・ 9ビット ・ ● ・ MSB ・ ● ・ ● ・ ● ・ ● ・ ● ● 55600 ・ (bps) (資差: 0.15%,許容量小: ・ ・ レバル3(低優先版位) ●	で」 コードの主成 レポー
ド グワック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × シ フェアコンボーネント設定 ドント ション ション シー マノアコンボーネント設定 シン ション シー メカ ション ション シー ション スクートアップ ション ション ション ション ション マートアップ ション ション ション ション マートアップ ション ション シー ション マー マー ション シー マー マー マー マー ション ション マー マー マー ション ジェスリック ジェスリック マー マー ジェスリック マー マー マー ジェスリック マー </td <td> 割り込み 割り込み 送信 受信 送信 受信 UART0クロック設定 動作クロック クロック・ソース デーク・ビット長設定 ○ 7ビット ③ 8ビット デーク転送方向設定 ④ 158 パリティ投定 ③ パリティビットなし ○ 0パリティ ストップ・ビット長設定 ジドの屋定です 受信データ・レベル設定 ● 非反転(通常) 転送レート設定 割り込み設定 受信完了割り込み設定(INTSR0) ビ エク-割り込み設定(INTSR0) </td> <td>CK00 「CLK/2^3 「ごごびいろ 9ビット 「MSB 「奇数パリティ 「偶数パリティ 「 方数 (□ (□ (□ (□ (□ (□ (□ (□ (□ (□ (□ (□ (□ (</td> <td>ら □-ドの主成 レポー</td>	 割り込み 割り込み 送信 受信 送信 受信 UART0クロック設定 動作クロック クロック・ソース デーク・ビット長設定 ○ 7ビット ③ 8ビット デーク転送方向設定 ④ 158 パリティ投定 ③ パリティビットなし ○ 0パリティ ストップ・ビット長設定 ジドの屋定です 受信データ・レベル設定 ● 非反転(通常) 転送レート設定 割り込み設定 受信完了割り込み設定(INTSR0) ビ エク-割り込み設定(INTSR0) 	CK00 「CLK/2^3 「ごごびいろ 9ビット 「MSB 「奇数パリティ 「偶数パリティ 「 方数 (□ (□ (□ (□ (□ (□ (□ (□ (□ (□ (□ (□ (□ (ら □-ドの主成 レポー
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × クェアコンボーネント設定 た シュースント 油 山 山 ファーンボーネント設定 シュースント シュースント シュースント シュースント スク マステム シュースント シュースント シュースント シュースント スク マステム シュースント シュースント シュースント シュースント スク マステム シュースント ション ション シュースント スク マステム シュースント ション ショ	割り込み ご 送信 受信 送信 受信 UART09Dック設定 動作クロック クロック・ソース データ・ビット長設定 〇アビット 第・10万イビットなし 〇のパリティ ストップ・ビット長設定 「パリティ設定です 受信完 「ラ・レベル設定 新送レート設定 新送レート設定 割り込み設定(INTSR0) ビーラー割り込み設定(INTSR0) コールパック機能設定	CK00 「には人2^3 「ご装数: 4000 kHz) 「ジゲト 「MSB 「可数パリティ (偶数パリティ 「「広坂 「(協友: 1000 kHz)) 「「「「「」」」」」 「(協友: 1000 kHz)) 「「「」」」」 「(協友: 1000 kHz)) 「「」」」 「(協友: 1000 kHz)) 「「」」」 「(協友: 1000 kHz)) 「「」」」 「(協友: 1000 kHz)) 「」」」 「(協友: 1000 kHz)) 「」」 「(協友: 1000 kHz)) 「」」 「(協友: 1000 kHz)) 「」」 「(協友: 1000 kHz)) 「」」 「(協友: 1000 kHz))	で] □-ドの主成 レポー!
ド クロック システム ユンボーネント 端子 citive_Touch_Project_Example.scfg × クロアコンボーネント設定 RXP 20 20 スカ 20 20 20 スカ 20 20 20 スカ 20 20 20 20 スカ 20 20 20 20 スカ 20 20 20 20 スクートアップ ジェネリック で Lobp 25 20 ドレフェア シスカート Config_LVD0 20 20 20 メ出力ボート Config_LORT0 21 21 21 21 ジェスリック で Lobg ア 21	割切込み 第 送信 送信 支信 送信 支信 UART000070股定 助作クロック クロック・ソース デーク・ビット長設定 ○ 7ビット ③ 8ビット デーク転送方向設定 ④ 10月イ・ビット長設定 ③ パリチィ・ビット長設定 ビット電データ・レバル設定 転送レート設定 転送レート設定 割切込み設定 受信完了割切込み設定(INTSRO) コールバック機能設定 ジ 受信完了	CK00 下にK/2^3 ・ 9ビット ・ ・ ・	で」 □-ドの主成 レポー

図 7-16 UART 通信コンポーネントの設定 (UARTO)



次に、[端子]タブを選択し、UART 通信機能に使用する端子を設定します。

本アプリケーションでは、UART0 (SAU00) に TOOLTxD, TOOLRxD を兼用する端子をそれぞれ割り当て ます。RL78/G22 では、以下の端子番号を設定します。

—RxD0	: 21
—TxD0	: 20

注意 本設定は使用するデバイスによって異なります。UART 通信機能を利用できる端子は、使用する FPBの回路図にてご確認ください。割り当てる端子番号は、使用する RL78 製品のユーザーズマ ニュアル ハードウェア編の「1.3 端子接続図」にてご確認ください。

ードウェアリソース	🗉 🖻 🎼 🦓	端子機能							⊘ ⊞ ⊑ ≥
フィルタ文字列を入力		フィルタフ	、力 (* = any :	string, ? = any	character)			व	べて
🏅 すべて	^	使用	機能	PIOR	端子割り当て	端子番号	方向	備考	コメント
\$⊯ I/Oポート			🐼 RxD0	PIOR1	P11/SI00/RxD0/TOOLRxD/SDA00/TS12/TI	/ 21		複数の端子機能が同一端子	
🗃 クロック発生回路		7 0	SCK00	PIOR1	/ 設定されていません	/ 設定されてい	なし		
> 🖏 タイマ・アレイ・ユニット			SCL00	PIOR1	✓ 設定されていません	/ 設定されてい	なし		
🖏 リアルタイムクロック			SDA00	PIOR1	/ 設定されていません	/ 設定されてい	なし		
> 🐗 クロック出力/ブザー出力制御回路			SI00	PIOR1	✓ 設定されていません	/ 設定されてい	なし		
强 A/Dコンバータ			SO00	PIOR1	✓ 設定されていません	/ 設定されてい	なし		
🗸 🌿 シリアル・アレイ・ユニット			🔕 TxD0	PIOR1	P12/SO00/TxD0/TOOLTxD/TS13/TI05/TO0:	/ 20 🤇 🔤		複数の端子機能が同一端子	
V O SAUO						•			
SAU00									
SAU01									
SAU02									
SAU03									
> SAU1									
> 階 シリアル・インタフェースIICA									
> 階 シリアル・インタフェースUARTA									
■ 割り込み機能									
響 〒−割り込み									
響 リセット機能									
■ 静雷容量式タッチヤンサ		1							

図 7-17 UART チャネル (UART0) の端子割り当て

使用するツールバージョンによっては UARTO の端子割り当てエラーが発生しますが、以下の理由から本 エラーは無視してください。

本アプリケーションノートでは、下記の端子をそれぞれ異なる用途で使用します。

TOOLRxD/TOOLTxD 端子: CS+で作成したプログラムを COM Port 経由で書き込むとき
 RxD0/TxD0 端子: スタンドアロン版 QE を用いたシリアル通信を行うとき

これらの端子機能は同一端子に割り当てられており、兼用機能となるため、スマート・コンフィグレータ 上では端子の競合エラーが出力されます。しかし、①および②を同時に使用することはありませんので、実 使用上は、端子は競合しません。よってこのまま使用して問題ありません。

▲ J)J/1/D-ジョンナエック ×	V 8 -
errors, 0 warnings, 0 others	
記述/説明 个	型
< ❷ 端子 (4項目)	
🤒 E04010003: RxD0 (Config_UART0で設定) が使用する端子と次の端子が競合しています : TOOLRxD (Systemで設定)、TOOLRxD (Pin Allocatorで設定).	端子
🤒 E04010003: TxD0 (Config_UART0で設定) が使用する端子と次の端子が競合しています : TOOLTxD (Pin Allocatorで設定)、TOOLTxD (Systemで設定).	端子
● E05000010: 端子20を複数の機能で使用できません。端子20にTxD0, TOOLTxDの機能が割り当てられています。	端子
◎ E05000010: 端子21を複数の機能で使用できません。端子21にRxD0, TOOLRxDの機能が割り当てられています。	端子

図 7-18 UART0 の端子割り当てエラー



7.5.4 LVD コンポーネント設定

ユーザ・オプション・バイト の電圧検出回路 0(LVD0)を設定します。

"LVD0"モジュールをクリックし、動作モードと検出電圧の設定を行います。リセット発生電圧 (VLVD0) を 2.62 (V)に設定します。

図 7-19 LVD コンポーネントの設定 (LVD0)

注意 RL78/G16 の場合は、電圧検出機能にセレクタブル・パワーオン・リセット回路を使用します。 本機能は [システム] タブからリセット発生電圧を設定します。設定内容を図 7-20 に示します。

JAN LABORAL		コードの生成 レポートの生成
3		
▼ オンチップ・デバッグ設定		
オンチップ・デバッグ動作設定 使用しない 	○ エミュレータを使う	
- Iミュレータ設定 ○ E2	E2 Lite	
疑似RRM/DMM機能設定 ○使用しない	● 使用する	
セキュリティID設定 □ セキュリティIDを設定する		
セキュリティID	0x00000000000000000000	
▼ セレクタブル・パワーオン・リセット回路		
RESET端子設定		
○使用しない	 使用する 	
動作モード設定		
リセット発生電圧(VSPDR)	2.84 × (V)	

図 7-20 リセット発生電圧の設定 (RL78/G16)



7.5.5 PORT コンポーネント設定
 LED に接続されるポートの設定を行います。
 本アプリケーション例では P62 と P63 を High レベル出力に設定します。

LED を制御するポートについては、使用するターゲットボードの回路図をご確認ください。

1. "PORT"モジュールをクリックし、"PORT6"にチェックを付けます。

#Capacitive_Touch_Project_Example.scfg	×	
ソノトウェアコンホーネント設定		コードの生成 レポートの生成
コンポーネント 🚵 🖾 🖧 🖻 🕀	設定	^
福禄 ものである。	ポート選択 PORT6	
 スタートアップ 、	PORTO PORT1	
 <i>c</i> r_bsp	PORT2 PORT3	
✓ ➢ 電源管理とリセット機能 ✓ Config IVD0	PORT4 PORT5	
✓ ▷ 入出力ポート		
Contig_PORI ✓ 龄 通信	PORT12 PORT13	
a Config_UART0 <	DPORT14	
✓	ポート・モード設定	
e r_ctsu	 Pmnレジスタ値を読み出す デジタル出力レベルを読み出す 	
		~
		>
概要 ボード クロック システム コンポーネント 端	子割り込み	

図 7-21 PORT コンポーネントの設定

2. 起動時にターゲットボード上の LED1, LED2 が点灯しないようにポートを設定します。
 [PORT6]タブをクリックし、"P62"と"P63"を High 出力に設定します。

ノノトリエノコノホ ホノト設定		コードの生成 レポートの生
コンポーネント 🚵 🛃 🖣	E 設定	
編 談 *	₹ #PORT6	
 	□ すべてに適用	
 in a state in a state	 ● 使用しない ○ 入力 ○ 出力 	1を出力
→ @ 電源管理とリセット機能		
 Config_LVDU 、 、 、		「花田刀
👔 Config_PORT		1を出力
Config_UART0		
 	P62 ○ 使用しない ○ 入力 ● 出力 ◆	☑ 1ē出力
er_ctsu	P63	
- m_couch	○ 使用しない ○ 入力 ⑧ 出力 🦛	☑ 1を出力

図 7-22 P62 と P63 を High 出力に設定



7.5.6 ボードサポートパッケージ

"r_bsp"モジュールを選択し、"Initialization of peripheral functions by Code Generator/Smart Configurator"が"Enable"に設定されていることを確認してください。

	√ onfigurations # Start up select # Control of illicit memory access detection(IAWEN) # Protected area in the RAM(GRAM1-0)	值 Enable (use BSP startup) Disable
 ● Mark B 45.9 £0705me ● Config_LVD0 ● 入出力ボート ● Config_PORT ● 通信 ● グロボロ(UART0) ■ ジロボリック ● ジエネリック ● ブエネリック ● ブエスリック ● ゴーCtsu ● rm_touch ■ 		Disabled Disabled Disabled Disabled Disables Enable Enable Disable Enable Disable Enable Disable Enable Disable Disable Enable Unused my.sw.warmstart.prec.function
	Watchdog Timer Initialize user function name Watchdog Timer setting user function name	my_sw_wdt_refresh_init_function my_sw_wdt_refresh_setting_function

図 7-23 r_bspの設定



7.6 未使用端子の設定

使用しない端子を Low レベル出力に設定することを推奨します。

本節では、例として"PORT41"を Low レベル出力に設定する手順を説明します。

注意 回路を作成する際は、端子処理などを適切に行い、電気的特性を満たすようにしてください。

1. "ポート"モジュールをクリックし、"PORT4"にチェックを付けます。

ソノトリエアコノハーイノト設定		コードの生成 レポートの生成
コンポーネント 🚵 🖬 🖓 🖂	B	1
	ポート選択 PORT4 PORT6	
 ◇ こ スタートアップ ◇ ジェネリック ◇ 「こちp ◇ ドライバ ◇ ご 下うすバ ◇ ご 電源管理とリセット機能 ⑦ Config_LVD0 ◇ ◇ 入出力ボート ⑦ Config_PORT ◇ ご ご ご ご ご ご ご ご ご ご ご ご ご ご ご ご ご ご ご	□ PORT0 □ PORT1 □ PORT2 □ PORT3 □ PORT4 □ PORT5 □ PORT6 □ PORT7 □ PORT12 □ PORT13 □ PORT14 □ □ PORT14 □ □ PORT12 □ PORT13 □ PORT14 □ □ PORT14 □	
💣 rm_touch		

図 7-24 ポートモジュールの設定

2. [PORT4]タブをクリックし、"P41"を Low 出力に設定します。

コンポーネント 泊山島	日 甲 設定	,
■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■	■ ポート選択 PORT4	
 ◇	□すべてに適用 ● 使用しない ○ 入力 ○ 出力 □ 内蔵ブルアップ □ TTL/(ッファ	1を出力
 ◇ 診ドライパ ◇ @ 電源管理とリセット機能 ◇ Config_LVD0 ◇ ロッボート 	P40 ● 使用しない ○ 入力 ○ 出力 ☑ 内蔵ブルアップ	1を出力
◇ (四) 八田 J 八一 P ② Config_PORT ◇ (金) 通信	P41 ○使用しない ○入力 ⑧出力 ◆ 載ブルアップ □ TTLバッファ	□ 1を出力
☞ Config_UART0 < ゆうまドルクエア < ゆうゴネリック ↓ c_tsu ↓ c_tsu ↓ m_touch		

図 7-25 P41 を Low 出力に設定



7.7 コード生成

スマート・コンフィグレータの⁵⁰アイコンをクリックし、コードの生成を行います。 コード生成時に注意文が表示されますが、無視してコード生成を続行してください。

【♂ コードの生成	×	
設定の競合やエラーにより、生成されたコードは実行時に問題が	ある可能性があります。	
無視してコード生成を続行しますか?		
□ 常に無視してコード生成しますか?		
	続行(P) キャンセル	

図 7-26 コード生成時の注意画面

スマート・コンフィグレータでオンチップ・デバッグ設定またはオプション・バイト設定を変更した場合、以下のメッセージが表示されることがあります。変更内容を確認した後に、[OK]をクリックします。

設定項目	古い値	新しい値
ユーザ・オプション・パイト値	-	EFFCE8
オンチップ・デバッグ・オプション・バ	-	84
デバッグ・モニタ領域の範囲	FE00-FFFF	OFE00-0FFFF
デバッグ・モニタ領域を設定する	No	Yes(Specify
[ОК	キャンセル

図 7-27 リンカオプション変更の確認

- 8. QE for Capacitive Touch の設定
- 8.1 QE for Capacitive Touch の起動
 - スタンドアロン版 QE (以下、QE) を起動します。
- 1. "QE-CapTouch (QE のインストールフォルダ) / eclipse / qe-captouch.exe"より QE を起動します。
- 2. 起動後の画面を以下に示します。

K−F•€=9× 🗖 🗖 🛱 🛱 🛱 🗮 🖤	メイン ステータス・チャート	ステータス・チャート			1 - D	パラメーター覧 × 🛛 🔯 🗊 🗇 🖱
モニタリングを有効にする モニタリング機能: 無効, 通信状態: 切断	^ ワークフロー⊠					
タッチI/F: ×	1. プロジェクトの準備	2. タッチインタフェースの準備	3. 調整	4. 実装と動作確認		
	 タッチインタフェースを使用するプロジェ クトを準備します。 	対象とするタッチインタフェースを準備し ます。	各タッチセンサについて自動調整処理を 行います。	タッチインタフェースの動作確認と撤調整 を行えます。		タッチI/F:
ボード・モニタ < ///チンファクス・チャート ×	 ★ ★<td></td><td>アロクラムの先们 Weight 2014年3月1日日本 Weight 2014年3月1日本 Weight 2014年3月1日本</td><td>10:27_0.05 LB 20:27_0.05 LB 30:45 LB 20:27_02 LB 20:28_02 LB 20:2</td><td></td><td>項目 値 パラメーター覧</td>		アロクラムの先们 Weight 2014年3月1日日本 Weight 2014年3月1日本 Weight 2014年3月1日本	10:27_0.05 LB 20:27_0.05 LB 30:45 LB 20:27_02 LB 20:28_02 LB 20:2		項目 値 パラメーター覧
	チューニング					
	✓ タッチインタフェース構成: <なし>	\$	ッチインタフェース構成			
9149	メソッド 種別 名前 タッチセン!	サ 寄生容量[pF] センサドライブパルス)	周波数[MHz] しきい値 計測時間[m	s] オーバーフロー		
2766 マルチ・ステータス・ チャート	עיעב		コンソール		1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	

図 8-1 QE 起動後画面

全画面表示にした際に表示が崩れる場合は、Windows 設定よりディスプレイの"拡大と縮小レイアウト" を"100%"に設定してください。



8.2 プロジェクトの準備

タッチインタフェースを使用するプロジェクトを準備します。

QE 起動後画面中央にあるワークフロー図の "プロジェクトの準備"に従い、設定します。



図 8-2 ワークフロー図 (プロジェクトの準備)

- 1. "プロジェクトフォルダの選択"下の […] をクリックし、CS+で作成したプロジェクトフォルダを選択します。
- 2. "マイコンの選択"下の […] をクリックし、使用するマイコンを選択します。



図 8-3 プロジェクトの準備



対象マイコンの製品名	×
ファミリ名	RL78 ~
グループ名	RL78/G22 ✓
ピン数	48pin:G ✓
ROM 容量	64KB:E ∽
製品名	R7F102GGE
	OK キャンセル

図 8-4 マイコンの選択

"マイコンの選択"で、以下のようなエラーが出た場合は、QE のインストールフォルダの格納場所に問題 がある可能性があります。一度 QE を終了し、フォルダを C:¥Renesas フォルダ以下などに移動させたの ち、QE を起動させてください。

Internal Error	– 🗆 X
Internal Error Reason: java.lang.NullPointerException	
	OK Details >>

図 8-5 マイコンの選択のエラー



8.3 タッチインタフェースの準備

ワークフロー図の"タッチインタフェースの準備"に従い、設定します。



図 8-6 ワークフロー図 (タッチインタフェースの準備)

1. "構成の選択"下の 🚩 をクリックし、"タッチインタフェース構成の新規作成"を選択します。

1. プロジェクトの準備	2. タッチインタフェースの準備	3. 調整	4. 実装と動作確認
タッチインタフェースを使用するプロジェ クトを準備します。	対象とするタッチインタフェースを準備し ます。	各タッチセンサについて自動調整処理を 行います。	タッチインタフェースの動作確認と微調整 を行えます。
 田焼対象 田焼対象を表示します。 マイコン マロジェクトの作成 スマート・コンフィグレータを使用して、 対象とするプロジェクトを作成します。 フロジェクトのたの スマート・コンフィグレータを使用して、 対象とするプロジェクトを作成します。 アロジェクトフリアルグの選択 我をするプロジェクトがあるフォルダ を選択します。 C:¥CS+_Workspace¥(… P1-2-の選択 我をするマイコンを選択します。 	 増成の選択 タッチインタフェース構成を選択/作成します。 タッチインタフェース構成の新規作成 ワッチインタフェース構成の新規作の 調整用ファイルの出力 調整用ファイルを出力する 調整用ファイルを出力する ワログラムの仮実器 main0関数内に、タッチ用メイン関数を 呼び出す処理を実装します。 プロジェクトのどルF DEを使用して、対象プロジェクトのビ ルドを行います。 	アログラムの実行 DEで対象プロジェクトのデバッグを勝 からルボードにあき込まれたチューニン プログラムが動作します。 プログラムが動作します。 プログラムが動作します。 御整を開始する 御整を開始する ● アドバンスドモード(高度な) プログリンスドモード(高度な) 御聴を開始する ● アドバンスドモード(高度な) ● アドバンスドモード(高度な)	プログラムの実装 タッチインタフェースを使用したプログ ラムを実装&ビルドします。 デバッグの開始 対象プロジェクトのデバッグを開始し、 プログラムを実行します。 シリアル接続 シリアル通信によるモニタリング機能を 有効にします。 ボーレート ボート番号 自動 ↓ 接続 モニタリング用ビューを表示し、モニタ リング機能を有効にします。

図 8-7 タッチインタフェース構成の新規作成



2. "タッチインタフェース構成の作成"ウィンドウが開き、タッチインタフェースを配置する領域が表示され ます。

右側の"タッチ I/F"パネルから[ボタン]をクリックすると、カーソルがボタンを配置できる状態になり、配置領域でクリックするとボタンを配置できます。

2つのボタン (Button00/Button01) を配置し、[ESC]キーを押してボタンの配置を終了します。

同様に、"タッチ I/F"パネルから[スライダ (横方向)]をクリックして、スライダ (Slider00) を配置します。

タッチインタフェースの追加を終了すると図 8-8の状態になります。

タッチインタフェース構成のファイル名: Capacitive_Touch_Project_Example 構成(メソッド)の設定	構成の流用/再編集
說明:	
Slider00	ØyJ€I/F \$
	静電容量方式
	自己容量方式 >
Button00 Button01	<i>ж9</i> У
	スライダ (横方向)
	スライダ(縦方向)
	ホイール
	キーパッド
	3Dジェスチャ (Al)
	タッチパッド
	シールド端子
	温度補正端子
	容量センサ
	電流センサ
設定	診断コード用端子
タッチI/Fの設定 総抵抗の設定 割り付けTSxの解除	タッチI/Fの削除
	構成(メソッド)の確認 *

図 8-8 ボタンとスライダの配置

3. 各ボタンの名前とタッチセンサの割り当てを行います。

配置した[Button00]をダブルクリックし、表示された"タッチインタフェースの設定"ダイアログで、以下のように設定します。

— タッチセンサ : TS24

— 抵抗値[Ω] :560

抵抗値は、使用するターゲットボードのユーザーズマニュアル、または回路図をご確認ください。

注意 ボタンにタッチセンサを割り当てる際に TS 端子の表示が消えてしまう場合は、Windows 設定を変更 する必要があります。Windows のシステム設定からディスプレイの"拡大と縮小レイアウト"を"100%" に設定後、QE for Capacitive Touch を再起動してください。

	タッチインタフェース	スの設定	×
	ボタン(自己)		7
	名前	Button00	
_	タッチセンサ TS24	抵抗值[Ω] > 560 >	
	OK	キャンセル へ	ルプ(H)

図 8-9 タッチインタフェースの設定 (ボタン)



4. 同様にして[Button01]を以下のように設定します。

― タッチセンサ	: TS23
— 抵抗値[Ω]	: 560

5. 同様にして[Slider00]を以下のように設定します。

: TS20
: TS21
: TS22

— 抵抗値[Ω]

― タッチセンサ

:	TS21
:	TS22
:	560



図 8-10 タッチインタフェースの設定 (スライダ)

6. タッチインタフェースが設定されると、図 8-11 のように電極が緑色の表示へと変化します。このまま [作成] をクリックします。

■!! タッチインタフェース構成の作成	×
タッチインタフェース構成のファイル名: Capacitive_Touch_Project_Example 構成(メソッド)の設定	構成の流用/再編集
就明:	∕7wF1/F \$
Slider00	静電容量方式
TS20 TS21 TS22	自己容量方式 🗸 🗸
Button00 Button01	ボタン
T524 T523	スライダ(横方向)
	スライダ(縦方向)
	ホイール
	キーパッド
	3Dジェスチャ (AI)
	タッチパッド
	シールド端子
	温度補正端子
	容量センサ
	電流センサ
設定	診断コード用端子
タッチバトの設定 総抵抗の設定 割り付けTSxの解除	タッチI/Fの削除
	構成(メソッド)の確認 ¥

図 8-11 タッチインタフェース設定後の構成



7. "チューニング"パネルに"タッチインタフェースの構成"が表示されます。

ロッチインク	フェーフ構成・C	anacitive To	uch Project Evamo	lo				
///////////////////////////////////////	/1-/1円/0.00	apacitive_10	uci_Project_examp	ie -				
メソッド	種別	名前	タッチセンサ	寄生容量[pF]	センサドライブパルス周波数[MHz]	しきい値	計測時間[ms]	オーバーフロー
config01	ボタン(自己)	Button00	TS24	-	-	-	-	なし
config01	ボタン(自己)	Button01	TS23	-	-	-	-	なし
config01	スライダ	Slider00	TS20, TS21, TS22	-		-	-	なし
config01	スライダTS	(Slider00)	TS20	-	÷.	-	-	<u>.</u>
config01	スライダTS	(Slider00)	TS21	-	-	-	-	-
confia01	スライダTS	(Slider00)	TS22	-	-	-	-	-

図 8-12 タッチインタフェース構成とチューニングパネル

8. 調整処理に必要なファイルを出力するためのフォルダを作成してください。本アプリケーション例では "Capacitive_Touch_Project_Example/src"の下に、新規で "qe_gen"という名前でフォルダを作成してい ます。QE ワークフロー図中の[調整用ファイルを出力する]をクリックした後、出力先のフォルダに作成 したフォルダを選択してください。



図 8-13 "qe_gen"フォルダ新規作成

以下が出力されるファイルを含めたフォルダ構成です。

Capacitive_Touch_Project_Example	← CS+プロジェクトフォルダ (6.新規プロジェクトの作成
- src	章で設定した任意のプロジェクト名)
- smc_gen	
- qe_gen	← 新規作成フォルダ
- qe_touch_config.c	← 出力ファイル
- qe_touch_config.h	← 出力ファイル
- qe_touch_define.h	← 出力ファイル
- qe_touch_sample.c	← 出力ファイル



9. CPU および周辺ハードウェアの周波数を設定します。出力先のフォルダを選択すると、ダイアログが表示されますので、CPU/周辺ハードウェア・クロック周波数 (fCLK)を設定し[OK]をクリックします。

周辺モジュールクロック(PCLKBまたはPCLKL)の周波数	×
周辺モジュールクロック(PCLKBまたはPCLKL)の周波数[MHz] 32	
OK キャンセル ヘルプ(H)	

図 8-14 周辺モジュールクロックの周波数の設定

10. マイコンへ供給する電圧を選択します。表示された"マイコンへの供給電圧"ダイアログで、電圧値を設 定して[OK]をクリックします。 使用するマイコンの電気的特性をご確認ください。

EVDD がある MCU の場合は、VDD を EVDD に読み替えて設定を行ってください。

マイコンへの供給電圧(VDD)	×
マイコンへの供給電圧(VDD) [V] 5.0 ・ ・ ・ ・ ・ ・ ・ ・ 5.0 ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・	ださい。
計測電圧設定 通常電圧 🗸	
OK キャンセル ヘルプ(H)

図 8-15 マイコンへの供給電圧の設定

11. "QE for Capacitive Touch"ダイアログが表示されます。 ダイアログの内容は、QE 画面下部の"コンソール"パネルにも表示されます。

QE for Capacitive Touch	×
 IDEを起動して、対象プロジェクトを開いてください。次に、以下の操作を行ってください。 ・コンパイラオプションを設定します。 ・ "Define Preprocessor Symbol (-D)" またはこれに準ずるオプションに "QE_TOUCH_CONFIGURATION" を追加します。 - "Include Directories (-1)" またはこれに準ずるオプションにファイル出力したフォルダを追加します ・ "include Directories (-1)" またはこれに準ずるオプションにファイル出力したフォルダを追加します ・ "main()関数内に、タッチ用メイン関数を呼び出す処理を実装します。 	•

図 8-16 QE for Capacitive Touch ダイアログ

ערכב	
ファイルを出力しました。	^
C:¥CS+_Workspace¥Capacitive_Touch_Project_Example¥src¥qe_gen¥qe_touch_define.h	
C:¥CS+_Workspace¥Capacitive_Touch_Project_Example¥src¥qe_gen¥qe_touch_config.h	
C:¥CS+_Workspace¥Capacitive_Touch_Project_Example¥src¥qe_gen¥qe_touch_config.c	
C:¥CS+_Workspace¥Capacitive_Touch_Project_Example¥src¥qe_gen¥qe_touch_sample.c	
IDEを起動して、対象ブロジェクトを開いてください。次に、以下の操作を行ってください。	
・コンパイラオブションを設定します。	
- "Define Preprocessor Symbol (-D)" またはこれに準ずるオプションに "QE_TOUCH_CONFIGURATION" を追加します。	
- "Include Directories (-I)" またはこれに準ずるオプションにファイル出力したフォルダを追加します。	
・main()関数内に、タッチ用メイン関数を呼び出す処理を実装します。	~
<	>

図 8-17 コンソールパネル



12. コンパイル・オプションを設定します。CS+を開き、"プロジェクト・ツリー"から "CC-RL(ビルド ツール)"を選択します。プロパティ"の "共通オプション" タブのうち、"よく使うオプション (コンパ イル)"の下の "定義マクロ"を選択し、右側に表示された […] をクリックします。

🕐 🎽 💌 🔨 CC-RL のプロパティ	a .
apacitive Touch Project Example (プロジェクト) V ビルド・モード	
R7F102GGExFB (マイクロコントローラ) ビルド・モード	DefaultBuild
スマート・コンフィグレータ(設計ツール) すべてのビルド・モードのプロパティを一括して変更	いいえ
CC-RL (FILK-Y-IL)	
RITRシミュレータ(デパッグ・ツール)	RL78-S3]7(-cpu=S3)
積和演算にMACH,MACHU命令を使用する	いいえ
> July July H (所 (所 (所) - ル))	
リファイルの種類	実行形式(ロード・モジュール・ファイル)
71 ビルド・ツール生成ファイル クロス・リファレンス 情報を出力する	0.007
	XeuidModeName%
Smart Configurator	
版画にレベル Setunの公式に一般が2	ちんたい取用しない アバインタンオイルための
appendix 3,277,54(1/0)1-54,1/2	2010年17月1日 - 1770年17月 シュアテレィイングルード・パスロ1 シュアテレィイングルード・パスロ1
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓	A44 () (()
in r_osp ishnのインクルード・パス	追加のインクルード・パス [2]
P- r_config > システム・インクルード・パス	システム・インクルード・パス [0]
●····································	定義マクロ [0]
> 使用するライブラリ・ファイル	使用するライブラリ・ファイル[0]
出力フォルダ	%BuildModeName%
出力ファイル名	%ProjectName%abs
◇ よく使うオブション(ヘキサ出力)	
定義マクロ	- (- マー・マー・マー・マー・マー・マー・マー・マー・マー・マー・マー・マー・マー・マ
正確していくり口名をいくり口名にいて厳固力の形式(1つ9つ指定します。「三定義国川の部分は省略り記じ、省略した場合、定義国をにとします。
00/01/01/00/07/02/01/85/08/98	

図 8-18 定義マクロの選択

13. 表示された "テキスト編集"ダイアログの"テキスト"フィールド内に"QE_TOUCH_CONFIGURATION" を追加し、[OK]をクリックします。

テナフト短伸	\sim
	~
テキスト(T):	
QE_TOUCH_CONFIGURATION	^
	~
< >	>
OK キャンセル ヘルブ()	H)

図 8-19 定義マクロの編集

14. サンプルプロジェクトへ"qe_gen"フォルダを追加します。CS+の"プロジェクト・ツリー"にエクスプロー ラから"qe_gen"フォルダをドラッグアンドドロップして追加します。



図 8-20 qe_gen フォルダの追加

15. "qe_gen"フォルダのインクルード・パスを追加します。プロパティ"の"共通オプション"タブのうち、"よく使うオプション (コンパイル)"の下の "追加のインクルードパス"を選択し、右側に表示された[…]をクリックします。

表示された "パス編集" ダイアログの "パス" フィールドに "src¥qe_gen" が追加されていることを確認して、[OK]をクリックします。

◇ よく使うオブション(エンパイル)		
最適化レベル	デバッグ優先(-Onothing)	
> 追加のインクルード・パス	追加のインクルード・パス[14]	
> システム・インクルード・パス	システム・インクルード・パス[0]	
> 定義マクロ	定義マクロ[1]	

図 8-21 コンパイラのインクルードパスの追加

J	パス編集		\times
1	ペス(1行につき1つのパス)(P):	a.	
	<mark>src¥ge_gen)</mark> src¥smc_gen¥r_pincfg src¥smc_gen¥r_ctsu src¥smc_gen¥r_config src¥smc_gen¥r_bsp¥mcu¥rl src¥smc_gen¥r_bsp¥mcu¥rl src¥smc_gen¥r_bsp¥mcu¥al	78_g22¥register_access¥ccrl 78_g22 I	
1	src#smc ∉en¥r bsb¥board¥i ≪	generic r1/8 g22	>
	参照(B)] 存在しないパスを許可する] 参照ボタンからパスを追加 パレースホルダ(L):	(N) 時に、サブフォルダも含める(S)	
	プレースホルダ ActiveProjectDir ActiveProjectMicomName ActiveProjectName BuildModeName Main ProjectDir く	値 C:¥CS+_Workspace¥Capacitive_Tou R7F102GGExFB Capacitive_Touch_Project_Example DefaultBuild C:¥CS+_Workspace¥Capacities_Tou	ich_Pro
		OK キャンセル ^	Jルプ(H)

図 8-22 追加したインクルードパスの確認



16. C ソース・ファイルの言語を選択します。

"コンパイル・オプション"のタブにある "ソース"をクリックし、"C ソース・ファイルの言語"をクリックします。右側に表示された ✓ をクリックし、"C99 (-lang=c99) "を選択します。

Cソース・ファイルの言語	C99(-lang=c99)	
C++ソース・ファイルの言語	C++14(-lang=cpp14)	
◇ 品質向上関連		
スタック破壊検出を行う	いいえ(オプション指定なし)	
不正な間接関数呼び出しを検出する	いいえ	
> メモリ・モデル		
> C言語		
> 文字コード		
> 出力コード		
> 出力ファイル		
> PESTA-UAF		
> MDRA-Gルール使宜		
200		
/ CONE		
Cソース・ファイルの言語		
Cソース・ファイルの言語を選択します。		
corロマンドの-langオブションに相当します。		

図 8-23 C 言語の規格の選択

- 17. オンチップ・デバック・オプション・バイトとユーザ・オプション・バイトを設定します。 "リンク・オプション"のタブにある"デバイス"をクリックし、以下を設定します。設定するオプショ ン・バイト値は、使用するマイコンのユーザーズマニュアルをご確認ください。
- オンチップ・デバック・オプション・バイト制御値:84
- デバック・モニタ領域を設定する
- ユーザ・オプション・バイト値

: はい (範囲指定) (-DEBUG_MONITOR=<アドレス 範囲>) : EFFCE8

オンチップ・テハック・オフション・ハイト制御値 デバッグ・モニタ領域を設定する	■111 84 (加) 100 (100 monitor=<アドレス範囲>) (100 monitor= 2</th
デバッグ・モニタ領域の範囲	OFEOD-OFFFF
ユーザ・オブション・バイトを設定する	(IC)(-USER_OPT_BYTE)
ユーザ・オフジョン・ハイト10	
201	
素教/園教配書情報	
> 変数/関数配置情報 セクション	
> 支数/ 関数配置情報 > セクション - ベリファイ	
 支数/側数配置情報 セクション ペリファイ メッセージ 	
 支数/動数記書情報 セクシュン セクション マリファイ メッセージ マの色 	
 支数/国数配置情報 セクション セクション メリンテイ メッセージ その他 	
 支数/国数配置情報 セクランシ ペリファイ メラセージ その色 パッグ情報を出力する パック情報を出力する パック情報を出力する パック情報を出力するかどうかを選択します。 	

図 8-24 オプション・バイトの設定

- CC-RL 無償評価版の V1.12.00 以降のバージョンを使用してコンパイルする場合は、コンパイラの最適 化レベルを"デバッグ優先 (-onothing)"を設定してビルドしてください。 "コンパイル・オプション"のタブにある"最適化"をダブルクリックし、"最適化レベル"に"デバッグ優先 (onothing)"を選択します。
- 備考 本操作はチューニング時のみ必要となります。チューニング後は、任意の最適化設定で使用可能で す。

最適化レベル	デバッグ優先(-Onothing)	×
✓ 最適化(詳細) ★ (注意) → ([interval → (interval → (
未使用static関数の削除を行う 開きたち見る問題をなるとは思う。	最適化レベルに合わせる(オブション指定なし)	
関設未進の関設呼び出したbr命令を使用する	戦い国ビレベルに合わせる(オフション指定なし)	
へ戦戦週にを行う またまた 見きひまだき	5.6.72	
ホインダ恒示売の空を考慮した範囲化を行う	10072	
モンユール間蔵運作用何加情報を出力する	0.0.12	
× 7970€X		~
最適化レベル		
コンパイルの最適化レベルを選択します。		
CCUTATION_0412371548309499		

図 8-25 コンパイラの最適化レベルの設定

19. main()関数内に、タッチ用メイン関数を呼び出す処理を実装します。main()関数から qe_touch_main()関数をコールします。

"main.c"ファイルへ以下のコードを追加します。

- extern void qe_touch_main(void);
- qe_touch_main();



図 8-26 main.c



20. "Config_UART0_user.c"にシリアル通信用の関数を追加します。

- 以下のコードをそれぞれ追加します。
- extern void touch_uart_callback(uint16_t event);
- touch_uart_callback(0);

Г

— touch_uart_callback(1);

52 53 54	/* Start user code for global. Do not edit comment generated here */ <u>extern void touch_uart_callback(uint16_t event);</u> /* End user code. Do not edit comment generated here */
74 75 76 77 78 79	<pre>static void r_Config_UARTO_callback_sendend(void) { /* Start user code for r_Config_UARTO_callback_sendend. Do not edit comment generated here */ touch_uart_callback(0); /* End user code. Do not edit comment generated here */ }</pre>
87 88 89 90 91 92	<pre>static void r_Config_UARTO_callback_receiveend(void) { /* Start user code for r_Config_UARTO_callback_receiveend. Do not edit comment generated here */ touch_uart_callback(1); /* End user code. Do not edit comment generated here */ }</pre>

図 8-27 Config_UART0_user.c

21. CS+でプロジェクトをビルドします。CS+のメニュー下にある Transform アイコンをクリックし、ビルドします。エラーまたはワーニングなしで終了することを確認します。

ビルド時に下図のワーニング(W0511187)が出た場合は、本書 40 ページの図 8-25 を参考にコンパイラ最 適化設定を「デバッグ優先 (-onothing) 」に変更して、再ビルドしてください。



図 8-28 ビルド時のワーニング (W0511187)

8.4 調整

ワークフロー図の"調整"に従い、設定します。



図 8-29 ワークフロー図 (調整)

 使用するデバック・ツールを選択します。CS+の"プロジェクト・ツリー"で"デバッグ・ツール"を右ク リックし、"使用するデバッグ・ツール"の"RL78 COM Port(C)"を選択します。



図 8-30 デバッグ・ツールの選択

2. "デバッグ・ツール"のプロパティで"通信ポート"を設定します。 本アプリケーション例では、COM24 を使用します。

~

図 8-31 デバッグ・ツールのプロパティ

通信ポートの COM 番号は、デバイス マネージャーで確認できます。

_ デバイス マネージャー	-	\times
ファイル(F) 操作(A) 表示(V) ヘルプ(H)		
> 🔲 プロセッサ		-
~ 📮 ポート (COM と LPT)		
Intel(R) Active Management Technology - SOL (COM3)		
💭 USB Serial Port (COM24)		
> 😰 ほかのデバイス		
> 🕕 マウスとそのほかのポインティング デバイス		
> 🛄 モニター		
> 🏮 ユニバーサル シリアル バス コントローラー		

図 8-32 デバイス マネージャー

- 3. COM Port デバック回路を有効にします。ターゲットボードの QE シリアル接続切り替えジャンパ (J16) がショートされていることを確認してください。
- ビルドとプログラムの書き込みを行います。PC とターゲットボードが USB ケーブルで接続されている ことを確認し、CS+の アイコンをクリックします。プログラム書き込み後のダウンロードが完了したら アイコンをクリックしてプログラムを停止させ、続いて アイコンをクリックして切断します。
- 5. QE のシリアル接続機能を実行します。デバック・ツールから切断後、PC とターゲットボードを接続している USB ケーブルを外し、QE シリアル接続切り替えジャンパ (J16) をオープンにします。 この後に QE と接続するため、PC とターゲットボードを USB ケーブルで再度接続します。この時に ターゲットボードは、書き込んだプログラムが動作し、QE との接続待機状態となります。

QE シリアル接続切り替えジャンパ (J16) は、ターゲットボードのユーザーズマニュアルをご確認ください。また、USB ケーブルはデータ転送対応のケーブルをご使用ください。



6. QE で"シリアル接続"の"ボーレート"を 7.5.3 節で設定した値に設定します。



図 8-33 ボーレートの設定

7. [調整を開始する]をクリックし、自動チューニングを開始します。



図 8-34 自動チューニング

8. 表示されたダイアログで"ボーレート"を設定し、[接続]をクリックします。

■ COM ポートへの接続	売(シリアル通信)	×
ボーレート 153600 📢		
COM ポート 自動		~
接続	キャンセル	ヘルプ(H)

図 8-35 ボーレートの設定

9. 続いて表示されたダイアログで CPU/周辺ハードウェア・クロック周波数を設定し、[OK]をクリックします。

周辺モジュールクロック(PCLKBまたはPCLKL)の周波数	×
周辺モジュールクロック(PCLKBまたはPCLKL)の周波数[MHz] 32	
OK キャンセル ヘルプ(H)	

図 8-36 周辺モジュールクロックの周波数の設定

10. 自動チューニングが開始されます。チューニングプロセスをガイドする[自動調整処理中]ダイアログを 適宜確認し、ダイアログ中の指示に従い操作を進めていきます。

■ 自動調整処理中	×
1/8: 調整処理を開始するための準備中です。 評価ボードは絶縁物などの上に置いてください。鉄板などの上に直接置くと正しく言 整中は、指示があるまでターゲットボード上のタッチセンサに触れないでください。	†測できません。 調
	キャンセル

図 8-37 自動調整処理中ダイアログ



いくつかのプロセスを経て、以下のようなダイアログが表示されます。

ここでは、タッチ感度を計測します。ダイアログで表示されている Button01 を通常の圧力でタッチしま す。タッチセンサに触れているとき、バーグラフは右に増加し、数値で示すタッチカウント値が大きくなり ます。

タッチセンサに触れたまま、PC キーボードのいずれかのキーを押して計測を確定します。



図 8-38 タッチ感度の計測 (ボタン)

- 11. もう一方のタッチセンサについても同様に計測します。
- 12.スライダのタッチセンサのタッチ感度を計測します。ターゲットボード上にあるスライダを通常の圧力 で上下または左右に 3~4回なぞったあと、PC キーボードのいずれかのキーを押して計測を確定しま す。

■ 自動調整処理中	×	
7/8:スライダの感度を計測します。(config01) ターゲットボード上にあるすべてのスライダを上下または左右に3~4回なぞったあと、キー を押してください。	ボードで何かキー	
	キャンセル	

図 8-39 タッチ感度の計測 (スライダ)



13.チューニングが完了すると、以下のようなダイアログが表示され、しきい値を確認できます。このしきい値はミドルウェアでタッチのイベント判定で使用されます。
 [調整処理の継続]をクリックします。これでチューニングは終了です。

リトライ対象の選択	メソッド	種別	名前	タッチセンサ	しきい値	オーバーフロー	警告/Iラ-		
	config01	ボタン	Button00	TS24	2426				
	config01	ボタン	Button01	TS23	2569				
	config01	スライダ	Slider00	TS20, TS21, TS22	2178				

図 8-40 タッチセンサのしきい値

14.[ファイルを出力する]をクリックし、調整結果が反映されたパラメータファイルを出力します。ファイルの出力先フォルダは 8.3 節で新規作成した"qe_gen"を選択し、ファイルを上書きします。
 出力されるファイル群は、8.3 節の[調整用ファイルを出力する]で出力された以下のファイル群と同じファイル名です。
 |- qe_touch_config.c
 ← 出力ファイル

- qe_touch_config.h	← 出力ファイル
- qe_touch_define.h	← 出力ファイル
- qe_touch_sample.c	← 出力ファイル

1. プロジェクトの準備	2. タッチインタフェースの準備	3. 調整	4. 実装と動作確認
タッチインタフェースを使用するプロジェ クトを準備します。	対象とするタッチインタフェースを準備し ます。	各タッチセンサについて自動調整処理を 行います。	タッチインタフェースの動作確認と微調整 を行えます。
 開発対象を表示します。 マイコン マロジェクトの作成 スマート・コンフィグレータを使用して、 対象とするプロジェクトを作成します。 大・スマート・コンフィグレータを使用して、 対象とするプロジェクトを作成します。 た、スマート・コンフィグレータで タッチセンサの設定とドライバの追加を 行います。 アロジェクトフォルダの選択 数とするプロジェクトがあるフォルダ を選択します。 C:¥CS+_Workspace¥(マイコンの選択 対象とするマイコンを選択します。 R7F102GGE 	 増成の選択 Aッチインタフェース構成を選択/作成 Lotalitive_Touch_Proje ↓ 借成を編集する 増成の選択へのより、 間整用ファイルの出力 御整用ファイルを出力する 御整用ファイルを出力する 御整用ファイルを出力する プログラムの変要表 アングラムの変要表 アングラムの変更 	<u>フログラムの実行</u> UFで対象ブロジュクトのデバッグで開 対・ブログラムを実行します。ター プットボードに書き込まれたチューニン ファクラムが動作します。 加整を開始する 小 アドパンスドモード(高度な: 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10	 プログラムの実装 ダッチインタフェースを使用したプログラムを実装をビルドします。 デバッグの開始 対象プロジェクトのデバッグを開始し、 プログラムを実行します。 シリアル通信によるモニタリング機能を 有効にします。 ボーレート 153600 ボート番号 自動 接続 モニタリング用ビューを表示し、モニタ リング機能を有効にします。

図 8-41 パラメータファイルの出力



8.5 実装と動作確認

8.5.1 モニタリング

ワークフロー図の"実装と動作確認"に従い、モニタリングを実施します。



図 8-42 ワークフロー図 (実装と動作確認)

- COM Port デバック回路を有効にします。PC とターゲットボードを接続している USB ケーブルを外し、 QE シリアル接続切り替えジャンパ (J16) をショートします。次に CS+と接続するため、PC とター ゲットボードを USB ケーブルで再度接続します。
- ビルドとプログラムの書き込みを行います。CS+の アイコンをクリックし、ビルドとプログラムの書き込みを行います。プログラム書き込み後のダウンロードが完了したら、 アイコンをクリックしてプログラムを停止させ、続いて アイコンをクリックして切断します。
- QE のシリアル接続機能を実行します。デバック・ツールから切断後、PC とターゲットボードを接続している USB ケーブルを外し、QE シリアル接続切り替えジャンパ (J16) をオープンにします。この後に QE と接続するため、PC とターゲットボードを USB ケーブルで再度接続します。この時にターゲットボードは、書き込んだプログラムが動作し、QE との接続待機状態となります。



 [接続]をクリックして、ターゲットボードとシリアル接続します。図 8-43 の赤枠内の表示が[接続]から [切断]に切り替わります。



図 8-43 シリアル接続

4. QE 画面の左上にある"ボード・モニタ"パネルの[モニタリングを有効にする]をクリックします。"モニタ リング機能: 無効"が"モニタリング機能: 有効"に切り替わります。

ボード・モニタ ×		ポード・モニタ ×	
モニタリングを有効にする モニタリング機能:無効	通信状態: シリアル通信 (UART / USB) で接続中 ^	モニタリングを有効にする モニタリング機能:	有効 通信状態: シリアル通信(UART / USB)で接続中 ^
9yfl/F:	~	タッチI/F:	~
	^		^
Slider00		Slider00	
~~~		→	
Button00 Button01		Button00 Button01	
	×		
<	>	<	>

図 8-44 モニタリングの有効化

タッチセンサに触れると、その状態が指のアイコンで表されます。



図 8-45 タッチセンサに触れた状態の表示



- 6. タッチカウント値をステータス・チャートヘグラフィカルに表示します。
   A. [ステータス・チャート]タブをクリックします。
  - B. 表示されたステータス・チャート画面の"タッチ I/F"の ✓ をクリックし、タッチインタフェースを選択します。

グラフには実行中のタッチカウント値が表示されます。選択したタッチセンサに触れると、タッチカウント値がグラフ上で変化することを確認できます。

緑のラインは、しきい値を表し、rm_touch ミドルウェアはボタンが操作/タッチされているかどうかを判断 するために使用されます。

グラフ下部の赤い矩形は、タッチカウント値がしきい値を超えてタッチが検出されたことを表示しています。



図 8-46 タッチカウント値のグラフ表示 (ボタン)



図 8-47 タッチカウント値のグラフ表示 (スライダ)



7. 必要に応じて、SNR 値を測定します。

A. [ステータス・チャート]タブ上の"データ収集の開始"をクリックします。

メイン ステータス・チャート ×	🖸 61 62 62 63 🖬 🗖 🗖
タッチI/F: Button00 @ config01 V 図現状態を同期する	^
V/F種別:ポタン(自己), TS端子: TS24	
計測値: 12055 ベースライン: 12047 Lきい値: 2426 タッチON/OFF差分値: 8	
データ収集の開始	
	~
14473	
13856	
13242	
12628	
12014	

図 8-48 タッチオフ状態のデータ収集

B. データ収集設定を行い、"データ収集の開始"をクリックします。

タッチオフ状態を収集している間は電極に触れないでください。また、緑色のバーはデータの収集率を 表しています。緑色のバーが右端まで達すると。データの収集率は 100%となりタッチオフ状態の収集 は完了します。

	■」 テーダ収集設定	X	
	データ収集数: 1000	~	
	データ収集対象: 〇 タッチOFF時 🔘 タッチOFF時+	タッチON時	
	収集データ: 〇計測値	差分值 🚽 👘	
	確率: 0.1% (3	.09) ~	
	データ収集の開始	ンセル	
タッチI/F: Button00 @	config01 v	□ 選択状態を同期する	
I/F種別: ボタン(自己), TS	端子: TS24		
計測値: 11512	ペースライン: 11547 しきい値: 2426	タッチON/OFF差分値: -35	
データ収集の終了			

図 8-49 データ収集の開始



C. 同様の手順でタッチ ON 時のデータを収集します。指が電極に触れていることを確認してから、"データ 収集の開始"をクリックします。緑色のバーが右端まで達するとタッチオン状態の収集は完了します。

QE for	Capacitive Touch	×
0	続けて、タッチON時のデータを収集します。計測の用意ができましたら、データ クしてください。	収集の開始ボタンをクリッ
		ОК
タッチ1/1	Button00 @ config01	🗌 選択状態を同期する
I/F種別	: ボタン(自己), TS端子: TS24	
言十測個	E: 11538 ペースライン: 11538 しきい値: 2426 タッチON/O	PFF差分值: 0
7-91	又集の開始	

図 8-50 タッチ ON 時のデータ収集の開始

D. データ収集が完了すると SNR 値が表示されます。

■] 標準偏差の計測結果		×
ノイズ標準偏差[NT]: 17.0 平均値[NT]: 2	最小值: -48 最大	·值: 51
ノイズ標準偏差[T]: 36.7 平均値[T]: 4543	シグナル/値: 4541 SNR	?値: 27.35
SNRは確率3.09で計算しています。		
計測結果はQE-Touchフォルダ内に保存されています。		
	[	ОК

図 8-51 SNR 値



8. 複数のタッチセンサのタッチカウント値をマルチ・ステータス・チャートヘグラフィカルに表示します。

QE 画面の左下にある[マルチ・ステータス・チャート]タブにて、表示するタッチセンサを選択します。



図 8-52 マルチ・ステータス・チャート

9. 必要に応じて、パラメータを手動で調整します。 QE 画面の右側にある"パラメーター覧"パネルでパラメータを調整します。



図 8-53 パラメータの調整



10."モニタリング機能: 有効"の状態で [モニタリングを有効にする] をクリックして、モニタリングを終了 します。

ボード・モニタ ×		
モニタリングを有効にする	モニタリング機能:有効,通信状態:シリアル通信(UAR	「/USB)で接続中 [^]
タッチI/F:	~ · · · ·	
		×
Slider0	0	^
~ ~	~	
Button00 Bu	itton01	
		~
<		>

図 8-54 モニタリングを終了する

11.[切断]をクリックし、シリアル接続を切断します。

1. プロジェクトの準備	2. タッチインタフェースの準備	3. 調整	4. 実装と動作確認
タッチインタフェースを使用するプロジェ クトを準備します。	対象とするタッチインタフェースを準備し ます。	各タッチセンサについて自動調整処理を 行います。	タッチインタフェースの動作確認と微調整 を行えます。
<ul> <li>         H発対象を表示します。         マイコン ◇     </li> <li>         アロジェクトの作成         スマート・コンフィグレータを使用して、対象とするプロジェクトを作成します。         た、スマート・コンフィグレータを使用して、対象とするプロジェクトを作成します。         アロジェクトのアイグレータで、タッチセンサの設定とドライバの追加を付います。      </li> <li> <b>DTジェクトワクルダの選択</b>         対象とするプロジェクトがあるフォルダ         を選択します。      </li> <li> <b>C:¥CS+_Workspace¥(</b> </li> <li> <b>C+コンの選択</b> 対象とするマイコンを選択します。      </li> </ul>	<ul> <li> <b>樹成の選択</b>          みッチインタフェース構成を選択/作成          はま。</li></ul>	<ul> <li>プログラムの実行</li> <li>DEで対象プロジェクトのデバッグを構 始に、プログラムを実行します。ター グットボードに書き込まれたチューニン グットボービに書き込まれたチューニン グットボービー書い。数年のも物情します。</li> <li><b>別整の開始</b></li> <li>オフロクの指示に従い操作します。</li> <li>問整を開始する</li> <li>アドパンスドモード(高度なに</li> <li>の数結果からパラメータファイルを出力 します。</li> <li>アイルを出力する</li> <li>今部ドリガを使用する</li> <li>診断コードを使用する</li> </ul>	プログラムの実装         ダッテインタフェースを使用したプログラムを実装 ビルドします。         アバッグの開始         対象プロジェクトのデバッグを開始し、 プログラムを実行します。         シリアル通信によるモニタリング機能を 有効にします。         ボーレード         オート番号         日動         切断         モニタリング機能を有効にします。         モニタリング用ビューを表示し、モニタリング機能を有効にします。

図 8-55 シリアル接続を切断



8.6 フローチャート (ソフトウェアタイマ)

図 8-56 にソフトウェアタイマを使用したタッチ計測制御処理のフローチャートを示します。



図 8-56 ソフトウェアタイマを使用したタッチ計測制御処理



9. 応用例

9.1 ハードウェアタイマでのタッチ計測

本章では、ハードウェアタイマを使用したタッチ計測周期の実装例を説明します。本アプリケーション例 では、32 ビット・インターバル・タイマの8 ビット・カウンタ・モードによるインターバル・タイマ機能 を使用します。また、動作確認のため、タッチセンサ (ボタン) でのタッチ判定結果に応じてターゲット ボード上の LED を点灯あるいは消灯させます。本アプリケーション例では、タッチセンサ① (TS_B1) に指 が触れ、タッチ判定結果が ON になると、LED1 が点灯します。

アプリケーションノート本編の「7. スマート・コンフィグレータの設定」の内容に加えて、以下の設定 を行ってください。

- 備考 32 ビット・インターバル・タイマの代わりに、タイマ・アレイ・ユニットや 12 ビット・インターバ ル・タイマを使用することも可能です。
- 9.1.1 スマート・コンフィグレータの設定 (ハードウェアタイマ)
- スマート・コンフィグレータの [クロック] タブを選択し、インターバル・タイマに使用するクロックを 設定します。 本アプリケーション例では低速周辺クロック (fSXP)を使用します。また、XT1 発振回路のチェックを 外します。

7ロック設定					J-K	の生成 レポートの生成
動作モード:	高速メイン・モード2.1	(V)~5.5(V)	•			
			1			
✓ 高速オンチゥブ・オシ  周波数・	22	- (Mile)				
4U000 55425-5-	32	<ul> <li>(MH2)</li> </ul>			fIHF	,
(STOPT~ドからのリリー	38 m 3時れよび SNO07E3	ードへの移行時に高速オンチップ		~	32	(MHz)
発振器を起動するための	の設定があります。)				fMA 32	(MHz)
					FCLH	(
				_		000 (kHz)
中連オンチップ・オシ	v-5				fimp	(MHz)
周波数:		- (MHz)				
			分周器			
X1発振回路			x1 *		fMD	æ 🕕
動作モード;		-				(MHz)
周波数:	5	(MHz)				
発振安定時間:						
低速オンチャプ・オシレー	b		- ]		fiL 32	768 (kHz)
周波数:	32.768	(kHz)				
信息の動作業件はウォッチ	ドッグ・ライマが動作、ま	たはfSXPが修連オンチップ・オシ			fSX	P
D-258884					32	2.768 (kHz)
XT1発振回路			]		fSX	R 🕕
動作モード:				•	-	(kHz)
周波数:		(kHz)				
XT1発掘モード:		~				
供給モード;		- /時の供給許可 -				

図 9-1 クロックの設定



ハードウェアタイマを使用したタッチ計測および LED の制御に必要な周辺機能を追加します。
 [コンポーネント] タブを選択し、シクリックして"コンポーネントの追加"ダイアログを開きます。"インターバル・タイマ"モジュールと"ポート"モジュールを選択し、[次へ] をクリックします。
 続いて、選択したコンポーネントに対してリソースを設定します。本アプリケーション例では以下の設定で使用します。

コンポーネントの追加			$\times$
選択したコンボーネン 加します	トのコンフィ <b>ク</b> レーションを追		#
インターバル・タイマ			
コンフィグレーション名:	Config_ITL000		
動作:	8 ビット・カウンタ・モード		$\sim$
リソース:	ITL000		~
注意:			
16ビット・キャプチャ・モ ません。 gビット・モードITIを曲日	-ドITL000_ITL001は、16ビット・カウント・モードITL01 日オス提会 16ビットは15732ビッロにしたけ彼田でき	2_ITL013と一緒に使用で +++ 4	き ^ >
ポート			
コンフィグレーション名:	Config_PORT		
リソース:	PORT		~
(?)	< 戻る(B) 次へ(N) > 終	了(F) キャンt	セル

図 9-2 インターバル・タイマとポートのリソース設定

3. インターバル・タイマの設定を行います。"Config_ITL000"コンポーネントを選択して、以下のように設定します。

リフトウェアコンポーネント設定		コードの生成 レポートの生成		
コンポーネント 🚵 🖾 🔩 🖻 🗉	設定			١
<ul> <li>■ 認</li> <li>▼ 2/ルタ入力</li> <li>&gt; ジェスリック         <ul> <li>* こ ジェスリック             <li>* ご このfig_LVD0             <li>* シ タイマ             <li>* Config_LVD0             </li> <li>* シ スカボート             <li>* Config_DRT             </li> <li>* 逆 スリカボート             <li>* Config_UART0             </li> <li>* ジェスリック             <li>* ジェスリック             <li>* ご ネリック             </li> </li></li></li></li></li></li></li></li></li></li></li></ul> </li> </ul>	クロック設定 動作クロック (fITL0) クロック・ソース インターバル・タイマ設定 インターバル時間 割り込み設定 □ つンペアー致またはキャプチャ 上順位 ・ エ ック 外す	fSXP       fITL0/128       20       空了を検出 (INTITL)       レベル3(低優先順位)	ms ~	▶周波数:0.256 kHz) )值:19.53125)

図 9-3 Config_ITL000の設定



4. LED に使用する端子の設定を行います。"ポート"モジュールで"P62"を High レベル出力に設定します。



図 9-4 P62の設定

5. スマート・コンフィグレータの右上の 🗊 アイコンをクリックして、コードの生成を行います。

以降は「8. QE for Capacitive Touch の設定」に従い設定を行ってください。



9.1.2 フローチャート (ハードウェアタイマ)

図 9-5 にハードウェアタイマを使用したタッチ計測制御処理のフローチャートを示します。



図 9-5 ハードウェアタイマを使用したタッチ計測制御処理



```
9.1.3 サンプルコード (ハードウェアタイマ)
 ハードウェアタイマでのタッチ計測のプログラム実装例 (qe_touch_sample.c) を以下に示します。
 * FILE : qe_sample_sample.c
 * DATE : 2025-02-25
 * DESCRIPTION : CTSU2L Program for RL78
 * NOTE: THIS IS A TYPICAL EXAMPLE.
 #include "qe touch config.h"
 #include "Config ITL000.h"
 void R CTSU PinSetInit(void);
 void qe_touch main(void);
 uint64 t button status;
 #if (TOUCH CFG NUM SLIDERS != 0)
 uint16 t slider position[TOUCH CFG NUM SLIDERS];
 #endif
 #if (TOUCH CFG NUM WHEELS != 0)
 uint16 t wheel position [TOUCH CFG NUM WHEELS];
 #endif
 void qe_touch_main(void)
    fsp err t err;
    BSP ENABLE INTERRUPT();
    /* Initialize pins (function created by Smart Configurator) */
    R CTSU PinSetInit();
    /* Open Touch middleware */
    err = RM_TOUCH_Open(g_qe_touch_instance_config01.p_ctrl,
 g_qe_touch_instance_config01.p_cfg);
    if (FSP SUCCESS != err)
    {
      while (true) {}
    }
    ITLSO &= ~ 01 ITL CHANNELO COUNT MATCH DETECTE;
    R Config ITL000 Start();
    /* Main loop */
    while (true)
    {
       while ( 00 ITL CHANNELO COUNT MATCH NOT DETECTE == (ITLS0 &
  01 ITL CHANNELO COUNT MATCH DETECTE)) {}
       ITLSO &= ~ 01 ITL CHANNELO COUNT MATCH DETECTE;
```



```
/* for [CONFIG01] configuration */
      err = RM_TOUCH_ScanStart(g_qe_touch_instance_config01.p_ctrl);
      if (FSP SUCCESS != err)
      {
          while (true) {}
      }
      while (0 == g_qe_touch_flag) {}
      g qe touch flag = 0;
      err = RM_TOUCH_DataGet(g_qe_touch_instance_config01.p_ctrl,
&button status, slider position, NULL);
if (FSP SUCCESS == err)
      {
          /* TODO: Add your own code here. */
          if (0 != button_status)
          {
             P6 bit.no2 = 0;
          }
          else
          {
             P6 bit.no2 = 1;
          }
      }
  }
}
```



10. 参考ドキュメント

〇ユーザーズマニュアル

• RL78/G22 ユーザーズマニュアル ハードウェア編 (R01UH0978)

• RL78 ファミリ ユーザーズマニュアル ソフトウェア編 (R01US0015)

(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。)

○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。)

〇ユーザーズマニュアル:開発環境

RL78/G22 Fast Prototyping Board ユーザーズマニュアル (R20UT5121)
 (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。)

**Oアプリケーションノート** 

- 静電容量センサマイコン 静電容量タッチ導入ガイド (R30AN0424)
- RL78 ファミリ スタンドアロン版 QE を使用した静電容量タッチアプリケーションの開発 (R01AN6574)
- シリアルポートを使用した RL78 デバッグ機能 (R20AN0632)
- RL78 ファミリ CTSU モジュール Software Integration System (R11AN0484)
- RL78 ファミリ TOUCH モジュール Software Integration System (R11AN0485)
- 静電容量センサマイコン 静電容量タッチ電極デザインガイド (R30AN0389)
- •

• RL78 ファミリ QE と SIS を使用した静電容量タッチアプリケーションの開発 (R01AN5512) (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)

ホームページ

- ルネサス エレクトロニクスホームページ <u>https://www.renesas.com/</u>
- Fast Prototyping Board 関連ページ <u>https://www.renesas.com/fast-prototyping-board</u>
- QE for Capacitive Touch 関連ページ <u>https://www.renesas.com/qe-capacitive-touch</u>
- 静電容量センサユニット関連ページ <u>https://www.renesas.com/solutions/touch-key</u>



## 改訂記録

		改訂内容	
Rev.	発行日	ページ	ポイント
1.00	Mar.20.23	-	初版
2.00	May.28.25	1	要旨を更新
		1	QE for Capacitive Touch の説明を注として追加
		4	1. 概要 章 を追加
		4	本書の対応デバイスを修正
		4	表 2-1 を更新
		4	表 2-1 に SIS モジュールを追加
		4	表 2-3 動作確認条件 を追加
		5	2.1 QE for Capacitive Touch の機能 章を追加(内容
			は Rev.1.00 の 1.システム概要章と同等)
		5	図 2-1 を更新
		6	3.1 章に CS+とスマート・コンフィグレータのイン
			ストール手順を追加
			3.1 章のタイトルを"開発ツールのインストール手
			順"に変更
		6	スタンドアロン版 QE for Capacitive Touch をイン
			ストール時の注意事項を追加
		7	3.2 章に文章および注を追加
		7	表 3-1 ボードのジャンパ設定 を追加
		9	表 4-1 を更新
		10	本アプリケーションノートのチューニングおよびモ
			ニタリング時の通信方法を追記
		10	Rev1.00の7.3.3章内のモニタリング時の制限事項
			を 5.1 草に移動し、備考として記載
		10	
		10	5.1 章に付属サンブルコードの説明を追加
		10	表 5-1 付属のサンブルコードの概要 を追加
		11	表 5-2 を更新
		13	
		14	7.2 草に図 7-5 と EVDD 設定の手順を追加
		15	
		16	SIS モジュールのタウンロード方法について注意を
		47	
		17	RL/8/G16の電圧検出機能の設定方法について注意
		17 10	
		17-10	7.4 コンホーネント追加 単を追加し、使用する主
			「コンホーホントの追加子順と白リン」への設定内容
		17-18	図 7-10 図 7-11 図 7-12 を追加
		19-26	
		10 20	ント設定の変更 章を追加し、各コンポーネントの
			設定内容を集約して記載
		20	 付属サンプルコードでの CTSU コンポーネントの
			設定を更新
		20	回路作成時の注意事項を記載
		20	図 7-14 を更新



RL78 ファミリ	FPB ボードでスタンド	アロン版 QE を用いたタ	タッチアプリケーション開発
-----------	--------------	---------------	---------------

21	Touch コンポーネントの設定内容を図 7-15 中の表 コニ ムー	
22	252 音の文音再新	
22		
23		
24		
24		
25		
25		
25	図 7-22 P62 と P63 を High 出力に設定 を追加	
26	7.5.6 ボードサボートパッケージ 草を追加	
	(内容は Rev.1.00 の 7.6 コート生成 草の項目1と   同等)	
27	図 7-24, 図 7-25 を更新	
28	図 7-26 コード生成時の注意画面 を追加	
28	図 7-27 のユーザ・オプション・バイト値を修正	
33	8.3章 項目2の文章を更新	
33	8.3章 項目3に文章および注意を追加	
33	図 8-8 を更新	
34	8.3章項目6に文章を追加	
34	図 8-11 を更新	
35	8.3章 項目8の文章を更新	
35	図 8-13 "qe_gen"フォルダ新規作成 を追加	
35	フォルダ名の誤記を修正	
36	8.3章項目9に文章を追加	
36	8.3 章 項目 10 に供給電圧の設定に関する留意点を 追加	
36	図 8-15 を更新	
38-40	開発手順 (項目 14 から 18 まで) の変更	
	・項目 14 に qe_gen フォルダ追加方法の説明を移	
	・標準・数学ライブラリの設定内容及び図 8-21	
	(Rev.1.00 の P.35 に記載) を削除	
38-39	8.3 章 項目 14,15,17 の文章を更新	
39	ユーザ・オプション・バイト値を修正	
39	図 8-24 のユーザ・オプション・バイト値を修正	
43	図 8-31, 図 8-32 を更新	
43	8.4 章 項目 3,4,5 の文章を更新	
45	8.4 章 項目9の文章を更新	
47	図 8-40 を更新	
48	8.5.1章 項目 1,2,3の文章を更新	
49	8.5.1 章 項目4に文章を追加	
49	図 8-44, 図 8-45 を更新	
50	8.5.1章 項目6のステータス・チャートへ表示する	
	タッチインタフェースの設定方法を修正	
50	図 8-46, 図 8-47 を更新	
51		
E1	0.5.1 早 項日 / の A の 文 早 を 史 利	
51	図 8-48 を更新	
51	図 8-48 を更新       8.5.1 章 項目 7 の B にデータ収集時の留意点を追	



RL78 ファミリ	FPB ボードでスタンドアロン	[,] 版 QE を用いたタ	ッチアプリケーション開発
-----------	-----------------	-------------------------	--------------

51	図 8-49 データ収集の開始 を追加
52	8.5.1章 項目7のCの文章を更新
52	図 8-50 データ収集の終了 を追加
52-54	図 8-51, 図 8-52, 図 8-53, 図 8-54 を更新
54	ソフトウェアタイマのサンプルコード
	(Rev.1.00 の P.54, P55 に記載) を削除
55	図 8-56 の CTSU ポート初期化、CTSU モジュール
	初期化をイニシャライズ処理に変更
55	図 8-56 に図のタイトルを追加
56	9.1 章の文章を更新
56	9.1 章に応用例での開発時の留意点を追加
57	9.1.1 章 項目2の文章を更新
57	図 9-2 を更新
57	9.1.1 章 項目3の文章を更新
58	9.1.1 章に項目 5 以降の設定手順の補足を追加
59	図 9-5 を変更
	・CTSU ポート初期化、CTSU モジュール初期化を
	イニシャライズ処理に変更
	・LED 点灯処理部を LED 点灯処理に変更
59	図 9-5 に図のタイトルを追加
60	9.1.3 章に文章を追加
62	10 章に参考ドキュメント (R01AN5512) を追加
	(Rev.1.00 の要旨から移動)
62	参考ドキュメント (ホームページ) に Fast
	Prototyping Board 関連ページを追加



#### 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。

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

- 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。
- 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

## 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。

## 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 ロ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/