

# RA2L1 タッチキー評価キット

[QE CapTouch 編] マニュアル

-本書を必ずよく読み、ご理解された上でご利用ください





注意事項	1
概要	2
1. 本書で説明する範囲	3
1.1. CD フォルダ構成	
2 初期設定	Δ
	<del>،</del>
2.1. QE ツールのインストール	
2.2. ノロシェクトの新規作成	
2.3. FSP の設定	
2.3.2. Stacks $\not> J$	
2.3.3. Pins & J	
2.4. エミュレータ(デバッガ)の接続設定	23
3. QE CapTouch の使用	25
3.1. CapTouch の起動	
3.2. タッチキーインタフェースの作成	
3.2.1. 自己容量基板(S16A)	
3.2.2. 相互容量基板(D55A)	
3.3. タッチキーインタフェースの調整	
3.3.1. 自己容量基板(S16A)	
3.3.2. 相互容量基板(D55A)	
3.4. ソースコードの変更	
3.4.1. 自己容量基板(S16A)	
3.4.2. 相互容量基板(D55A)	
3.5. ビルド•実行	
3.6. モニタリング	
	10
4. まとめ	46
5. 付録	47
取扱説明書改定記録	
お問合せ窓口	





# 注意事項

本書を必ずよく読み、ご理解された上でご利用ください

# 【ご利用にあたって】

- 本製品をご利用になる前には必ず取扱説明書をよく読んで下さい。また、本書は必ず保管し、使用上不明な点がある場合は再読し、よく理解して使用して下さい。
- 2. 本書は株式会社北斗電子製マイコンボードの使用方法について説明するものであり、ユーザシステムは対象ではありません。
- 3. 本書及び製品は著作権及び工業所有権によって保護されており、全ての権利は弊社に帰属します。本書の無断複 写・複製・転載はできません。
- 弊社のマイコンボードの仕様は全て使用しているマイコンの仕様に準じております。マイコンの仕様に関しましては 製造元にお問い合わせ下さい。弊社製品のデザイン・機能・仕様は性能や安全性の向上を目的に、予告無しに変更 することがあります。また価格を変更する場合や本書の図は実物と異なる場合もありますので、御了承下さい。
- 5. 本製品のご使用にあたっては、十分に評価の上ご使用下さい。
- 6. 未実装の部品に関してはサポート対象外です。お客様の責任においてご使用下さい。

## 【限定保証】

- 1. 弊社は本製品が頒布されているご利用条件に従って製造されたもので、本書に記載された動作を保証致します。
- 2. 本製品の保証期間は購入戴いた日から1年間です。

## 【保証規定】

#### 保証期間内でも次のような場合は保証対象外となり有料修理となります

- 1. 火災・地震・第三者による行為その他の事故により本製品に不具合が生じた場合
- 2. お客様の故意・過失・誤用・異常な条件でのご利用で本製品に不具合が生じた場合
- 3. 本製品及び付属品のご利用方法に起因した損害が発生した場合
- 4. お客様によって本製品及び付属品へ改造・修理がなされた場合

# 【免責事項】

弊社は特定の目的・用途に関する保証や特許権侵害に対する保証等、本保証条件以外のものは明示・黙示に拘わらず 一切の保証は致し兼ねます。また、直接的・間接的損害金もしくは欠陥製品や製品の使用方法に起因する損失金・費用 には一切責任を負いません。損害の発生についてあらかじめ知らされていた場合でも保証は致し兼ねます。 ただし、明示的に保証責任または担保責任を負う場合でも、その理由のいかんを問わず、累積的な損害賠償責任は、弊 社が受領した対価を上限とします。本製品は「現状」で販売されているものであり、使用に際してはお客様がその結果に 一切の責任を負うものとします。弊社は使用または使用不能から生ずる損害に関して一切責任を負いません。 保証は最初の購入者であるお客様ご本人にのみ適用され、お客様が転売された第三者には適用されません。よって転 売による第三者またはその為になすお客様からのいかなる請求についても責任を負いません。 本製品を使った二次製品の保証は致し兼ねます。



概要

本書は、

「RA2L1 タッチキー評価キット」

でタッチキーのルネサスエレクトロニクス製ミドルウェアである QE for Cap Touch の使い方について解説を行うものです。

RA2L1 マイコンはルネサスの新しいタッチキーユニットである CTSU2 を搭載しています。

本書では、キット付属の2種類のタッチキー基板、「自己容量タイプタッチキー基板(S16A)」及び「相互容量タイプタ ッチキー基板(D55A)」を、QE for Cap Touch で使う方法を示しています。

QE for Cap Touch は、複雑なタッチキーユニットの初期設定や、タッチした際の閾値のチューニングや測定値のモニタ等が行える高機能なソフトとなります。

タッチキー(CTSU2)のプログラムをスクラッチで書き下す場合は、本書とは別なマニュアルで、タッチキー(CTSU2) のソフトウェア編マニュアルがありますので、そちらも必要に応じて参照ください。





# 1. 本書で説明する範囲

# 1.1. CD フォルダ構成

PROJECT/	RA/	RA2L1_QE_CAP_TOUCH_S.zip	RA2L1 向け e2studio プロジェクト (自己容量, S16A タッチパッド向け)
	RA/	RA2L1_QE_CAP_TOUCH_M.zip	RA2L1 向け e2studio プロジェクト (相互容量, D55A タッチパッド向け)

e2studioのプロジェクトのアーカイブとなっていますので、ワークスペースにインポートしてください。





# 2. 初期設定

## 2.1. QE ツールのインストール

タッチキーは、Renesas QE Cap-touch でサポートされる機能となりますので、e2studio に、Renesas QE Cap-touch をインストールしてください。

※[重要] Renesas QE Cap-touch を使って PC からタッチキー動作を見る場合は、エミュレータ(ルネサス E2Lite も しくは E2)が必要です (エミュレータがない場合は、CD に格納されている作成済みのプロジェクトを開き、タッチ動作 を確認することができます。新規にプロジェクトを作成する場合はデバッガが必須となります。)

Alk	プ(H)	
3	ようこそ(W)	
?	ヘルプ目次(H) たまの	
: *	(使業(E) ダイナミック・ヘルプ(D)	
	キー・アシスト(K) 虎の巻の表示(C)	Ctrl+シフト+L
	RA Helpdesk	
R	RenesasRulz コミュニティー・フォーラム	1
ø	Renesas ツールチェーンの追加	
~2	Perform Setup Tasks	
<i>e</i> 2	更新の検査	
<b>63</b>	新規ソフトウェアのインストール	
	Renesas e2 studio feedback	
2	IAR Embedded Workbench plugin manager	
8	e <sup>2</sup> studio について(A)	

ヘルプー新規ソフトウェアのインストールから、





インストール						×
利用できるソフトウェア						
インストールしたい項目をチェック					9	-
作業対象(W): QE Common Update Site - http://tool-support.renesas.com/e/	2studio/qe/qe-common/	~	追加( <u>A</u> )		管理( <u>M</u>	)
74ルタ入力				<u>ಕ</u> ಗ	て選択(	<u>S</u> )
名前	バージョン			濯択を)	すべて解	除(D)
✓ ✓ III Renesas QE						14(2)
☑ 🖗 Renesas QE common	1.7.0.v20191204-1122					
☑ @ Renesas QE for Capacitive Touch[RA, RL78]	1.3.0.v20210218-1154					
2 項目が選択されました。						
						â
☑ 利用できるソフトウェアの最新バージョンだけを表示(L)	□ 既にインストールされた項目を隠す(且)					
☑ カテゴリーで項目を分類( <u>G</u> )	<u>すでにインストール済み</u> なのは何?					
□ ターゲット環境に適用できるソフトウェアのみ表示						
□ 小車ないフトウェアを見つけるために インストール中に車新サイト全てに接续(^)						
?	< 戻る( <u>B</u> ) 次へ( <u>N</u> ) >		終了( <u>F</u> )	:	キャンセノ	٧

QE Common Update Site を選択して、 Renesas QE common Resnesas QE for Capacitive Touch をインストールしてください。





・ローカルインストールの場合

インターネットに直接接続していない場合は、ルネサスエレクトロニクスのサイトから zip ファイルをダウンロードして、ローカルファイルからインストールしてください。

All	プ(H)						
3	ようこそ(W)	1					
?	ヘルプ目次(H)						
22	検索(E)						
	ダイナミック・ヘルプ(D)	-					
	キー・アシスト(K)	Ctrl+シフト+L					
	虎の巻の表示(C)	-					
	Renesas Help	>					
	CMSIS Packs Management	>					
ø	Renesas ツールチェーンの追加						
~??	Perform Setup Tasks						
<i>°</i> 2	更新の検査						
<b>6</b> .	新規ソフトウェアのインストール						
	IAR Embedded Workbench plugin manager						
6	e <sup>2</sup> studio について(A)						

ヘルプー新規ソフトウェアのインストール

(雪 インストール			– 🗆 X
利用できるソフトウェア サイトを選択するか、またはサイトのロケーションを入力してください。			
作業対象( <u>W</u> ): <sup>8</sup> サイトを入力または選択		~ 追加( <u>A</u> )	管理( <u>M</u> )
フィルタ入力			すべて選択( <u>S</u> )
名前 ① 選択したサイトがありません。 詳細	パージョン		選択をすべて解除( <u>D</u> )
			ŵ
<ul> <li>✓利用できるソフトウェアの最新バージョンだけを表示(L)</li> <li>✓カテゴリーで項目を分類(G)</li> <li>□ターゲット環境に適用できるソフトウェアのみ表示</li> <li>✓必要なソフトウェアを見つけるために、インストール中に更新サイト全てに接続(C)</li> </ul>	□ 既にインストールされた項目を隠す(出) <u>すでにインストール済み</u> なのは何?		
?	< 戻る( <u>B</u> ) 次へ( <u>N</u> ) >	終了( <u>F</u> )	キャンセル

追加





📴 リポジトリーを追加	—		×
名前( <u>N</u> ): ロケーション( <u>L</u> ): http://		ローカル アーカイン	( <u>O</u> ) <sup>j</sup> ( <u>A</u> )
?	追加( <u>D</u> )	キャンセ	:Jl

アーカイブで、QE Cap Touch のアーカイブファイル(.zip ファイル)を選択。

※RenesasQE\_cap-touch\_RARL78SynergyRX\_V301.zip を展開し、<u>展開したフォルダに含まれる</u> RenesasQE\_cap-touch\_RARL78SynergyRX\_V301¥QE-CapTouch¥**RenesasQE\_cap-touch\_V301.zip** 上記太字の zip ファイルを開く(V3.01 の場合、バージョンは読み替えてください)



🖾 リポジトリー	を追加			×
名前( <u>N</u> ):			ローカル(	<u>0</u> )
ロケーション( <u>L</u> ):	jar:file:/C:/spool/RenesasQE_cap-touch_RARL78Syne	rgyR	アーカイブ	( <u>A</u> )
ОК				
?	追加( <u>D</u> )		キャンセ	JL

追加





P					
📴 インストール					×
利用できるソフトウェア インストールしたい項目をチェック					
作業対象(W): jar:file:/C:/spool/RenesasQE_cap-touch_RARL78SynergyRX_V301	I/QE-CapTouch/RenesasQE_cap-touch_V30 ∨	追加( <u>A</u> )		管理( <u>N</u>	<u>1</u> )
フィルタ入力			đ	べて選択	( <u>S</u> )
名前	パージョン		選択を	をすべて解	祥除( <u>D</u> )
> 🔽 💷 Renesas QE					
		>			
2項目が選択されました。					
詳細					
▽ 利用できるソフトウェアの最新パージョンだけを表示(1)	□ 時にインストールされた頂目を開す(日)				
ビーカルビビジン・シンクロス Min ( シンクビッビル Concest And )	すでにインストール済みなのは何?				
☑ 必要なソフトウェアを見つけるために、インストール中に更新サイト全てに接続(⊆)					
?	< 戻る( <u>B</u> ) 次へ( <u>N</u> ) >	終了(E)		キャンセ	IL

#### チェックを入れて次へ

③ 証明書 >	<
これらの証明書を信頼しますか?	
REE-SSD Eclipse; Broad-based Solution Business Unit; Renesas Electronics Europe Ltd.	
すべて選択( <u>S</u> ) 選択をすべて解除( <u>D</u> )	
<ul> <li>REE-SSD Eclipse; Broad-based Solution Business Unit; Renesas Electronics Europe Ltd. REE-SSD Eclipse; Broad-based Solution Business Unit; Renesas Electronics Europe Ltd.</li> </ul>	
詳細 & E <u>x</u> port	
⑦ 違択を承認(A) キャンセル	]

証明書関連は承認してください。





😰 Y7	トウェア更新	$\times$
?	e <sup>2</sup> studio を再起動してソフトウェアの更新を適用しますか?	
	すぐに再始動( <u>R)</u> いいえ( <u>N</u> )	

ー度 e2studio を再起動して、インストールを有効化してください。







# 2.2. プロジェクトの新規作成

#### e2studio(+FSP 環境)上で

6	works	pace_RA -	e <sup>2</sup> studio								
ファイ	<i>J</i> μ( <u>F</u> )	編集( <u>E</u> )	ナビゲート( <u>N</u> )	検索( <u>A</u> )	プロジェクト( <u>P</u> )	Renesa	s <u>V</u> iews 実行( <u>R</u> )	ウィンドウ( <u>W</u> )	へルフ	<sup>1</sup> ( <u>Н</u> )	
	新規(	N)			Alt+シフト+N	>	Renesas C/C++ F	Project	>	Renesas Debug	18
ファイルを開く(.)				C	C/C++ Project			Renesas RA	H		
	ファイノ	レ・システム	からプロジェクトそ	を開く		<b></b>	プロジェクト(R)			_	_
	最近(	のファイル				> 🖻	サンプル(X)			3	

#### ファイル - 新規 - Renesas C/C++ Project - Renesas RA

#### を選択。

📴 New C/C++ P	oject		_		×
Templates for F	enesas RA Project				
All C/C++	Renesas RA C/C Create an executal for Renesas RA.	++ <b>Project</b> ble or static library C/0	C++ proj	ject	
?	< 戻る( <u>B</u> ) 次へ( <u>N</u>	<b>)&gt;</b> 終了(E)		キャンセ	μ

#### 次へ。

Renesas RA C/C++ Project			×
Renesas RA C/C++ Project Project Name and Location			2
Project name RA2L1_QE_CAP_TOUCH			
☑ デフォルト・ロケーションの使用(D)   □ケーション(L): C¥Users¥win64-7¥Documents¥e2_studio¥workspace_RA¥RA2L1_QE_CAP_TOUCH   ファイル・システムを選択(Y): デフォルト ∨	al de constantes de la constante	参照( <u>R</u> ).	
You can download more Renesas packs here			
⑦ < 戻る(B) 次へ(N) > 終了(D)	4	キャンセノ	۶.

プロジェクト名を入力。







📴 Renesas RA	C/C++ Project			_		×
Renesas RA C Device and Too	/C++ Project Jls Selection					\$
- Device Selection FSP Version: Board: Device: Language:	an 3.5.0 Custom User Board (Any Device) R7FA2A1AB3CFM C O C++	Board Description Device Details TrustZone Pins Processor	No 64 Cortex-M23			
Toolchains GNU ARM Er 10.3.1.202108	mbedded	Debugger J-Link ARM				~
?		< 戻る( <u>B</u> ) 次・	へ( <u>N</u> ) > 終了(E	)	キャンセ	μ

Device の右側のボタンを押し、デバイス選択画面を開く。

8	— 🗆	×
Device Selection		
You can filter devices by regular expression		
Search Device		
Device	Pin	
✓ RA2		
> RA2A1		
> RA2E1		
> RA2E2		
✓ RA2L1		
✓ RA2L1 - 100 Pin		
R7FA2L1AB3CFP	100	
R7FA2L1AB2DFP	100	
R7FA2L1A93CFP	100	
R7FA2L1A92DFP	100	
> RA2L1 - 80 Pin		
> RA2L1 - 64 Pin		
> RA2L1 - 48 Pin		
> RA4		
> RA6		
(?)	OK ++v)	211

R7FA2L1AB2DFP を選択して「OK」。





📴 Renesas RA	C/C++ Project			_		×
Renesas RA C Device and Too	/C++ Project Jls Selection					2
Device Selection	on					
FSP Version:	3.5.0 ~	Board Description				
Board:	Custom User Board (Any Device) $\sim$					
Device:	R7FA2L1AB2DFP	Device Details				
Language:		TrustZone Pins Processor	No 100 Cortex-M23			
Toolchains		Debugger				
GNU ARM Er	nbedded	J-Link ARM None E2 (ARM) E2 Lite (ARM) J-Link ARM				~
?		< 戻る( <u>B</u> ) 次	∧( <u>N</u> ) > 終了( <u>F</u> )		キャンセ	λ

Debuggerを選択。(ここでは、E2 Lite(ARM)を選びます)

※QE Cap Tocuh で新規にプロジェクトを作成する場合、エミュレータ(デバッガ)が必要です (作成済みのプロジェクトを使って動作を見る場合は、UART(SCI)経由で情報のモニタが行えます→エミュレータは不 要です)

Renesas RA C/C++ Project	_		×
Renesas RA C/C++ Project Build Artifact and RTOS Selection			\$
Build Artifact Selection   Executable  Project builds to an executable file  Static Library	RTOS Selection No RTOS		~
Project Duilds to a static library Executable Using an RA Static Library Project builds to an executable file Project uses an existing RA static library project			
(?)	< 戻る( <u>B</u> ) 次へ( <u>N</u> ) > 終了(E)	キャンセ	μ

デフォルトのまま「次へ」。





Renesas RA C/C++ Project			Х
Renesas RA C/C++ Project Project Template Selection			\$
Project Template Selection			
Bare Metal - Minimal Bare metal FSP project that includes BSP. This project will initialize clocks, pins, stacks, and the C runtin [Renesas.RA.3.5.0.pack]	ne enviro	nment.	
Code Generation Settings			
Use Renesas Code Formatter			
? < 戻る(B) 次へ(N) > 終了(E	)	キャンセ	μ

Bare Metal - Minimal にチェックを入れ「終了」。





2.3. FSP の設定



FSP の設定を行う場合は、e2studio ウィンドウ右上のビュー設定を、「FSP Configuration」を選択してください。 ※他のビュー設定の場合は、プロパティが表示されません

※画面レイアウトが崩れて、どこに何があるか判らなくなった場合は、



ウィンドウ - パースペクティブ - パースペクティブのリセット

を実行して、画面レイアウトを初期状態に戻してください。

#### 2.3.1. Clocks タブ

Clocks Configuration	Generate Project Content
	🔣 Restore Defaults
1¢00000 → ICLK Div /1 → ICLK 48MHz	
LOCO 32768Hz → PCLKB Div /2 → PCLKB 24MHz	
MOCO 8MHz Clock Src: HOCO V PCLKD Div /1 V PCLKD 48MHz	
SUBCLK 32768Hz	
HOCO 48MHz V	
$\sim$ CLKOUT Disabled $\sim$ $\rightarrow$ CLKOUT Div /1 $\sim$ $\rightarrow$ CLKOUT 0Hz	
Summary BSP Clocks Pins Interrupts Event Links Stacks Components	

XTAL の値を 16MHz(16000000)に変更する(※タッチキーでは XTAL は使っていないので必須ではありません)。





# 2.3.2. Stacks タブ

Stacks Configuration		Generate Project Content
Threads 🕘 New Thread 🔊 Remove 📄	HAL/Common Stacks	New Stack > 🐣 Extend Stack > 🖌 Remove
♥ ☆ HAL/Common	<ul> <li> <i>g_ioport I/O Port</i>         (r_ioport)         <ul> <li></li></ul></li></ul>	
Objects 🐑 New Object > 🔊 Remove		
Summary BSP Clocks Pins Interrupts Event Link	ks Stacks Components	



New Stack – CapTouch – Touch(rm\_touch) を追加。



![](_page_18_Picture_0.jpeg)

Stacks Configuration				Generate Project Content
Threads 🕢 New Thread 🔬 Remove 📄	HAL/Common Stacks		🗿 New Stack > 👙	🛓 Extend Stack > 🛛 🙀 Remove
<ul> <li>✓ MAL/Common</li> <li>♥ g_ioport I/O Port (r_ioport)</li> <li>♥ Touch (rm_touch)</li> </ul>	<pre>g_ioport I/O Port     (r_ioport)</pre>	U Touch (rm_touch)	)	
		CTSU (r_ctsu) Add DTC Driver for Transmission [Recommended but optional] (3)	Add DTC Driver for Reception [Recommended but optional] (4)	4dd SCI UART Driver for monitor of QE (5)
Objects 🕢 New Object > 🔊 Remove				
Summary BSP Clocks Pins Interrupts Event Links Stacks	Components			

(1)~(5)のスタックが追加されます。

## (1)Touch(rm\_touch)の設定

🔲 プロパテ	イー 🗙 問題 🦓 スマート・ブラウザー		<b>1</b> 8 <b>1</b>
Touch (r	m_touch)		
Settings	プロパティ × Common	値	
APIInto	Parameter Checking	Default (BSP)	
	Support for QE monitoring using UART	Enabled	
	Support for QE Tuning using UART	Enabled	×
		Enabled	
		Disabled	

Support for QE monitoring using UART : Enabled に変更 Support for QE Tuning using UART : Enabled に変更

![](_page_18_Picture_6.jpeg)

![](_page_19_Picture_0.jpeg)

#### (2)CTSU(r\_ctsu)の設定

□ ブロパティー 🛛 🔝 問題 🎭 スマート・ブラウザー				
CTSU (r	CTSU (r_ctsu)			
Settings APL Info	プロパティ ✓ Common	值		^
Arrino	Parameter Checking	Default (BSP)		-
	Support for using DTC	Enabled		
	Interrupt priority level	Priority 2		
	✓ Module CTSU (r_ctsu)			
	Scan Start Trigger	Software		
	✓ Pins			
	TSCAP	<unavailable></unavailable>		
	TS00	<unavailable></unavailable>		
	TS02-CFC	<unavailable></unavailable>		~

Support for using DTC : Enabled に変更(※DTC を使うかは任意です)

HAL/Common Stacks	💽 New Stack > 🚔 Extend Stack > 🖟 Remove	
g_ioport I/O Port (r_ioport)	Journa (rm_touch)	
۵.	0	
	CTSU (r_ctsu)	
	☆ Add DIC Driver for Transmission [Recommended but optional] New → ① Transfer (r_dtc) Add DTC Driver Add DTC Driver のボックス上で「左」クリッ	ク
	(2)で DTC を有効化しない場合は (3)(4)はスキップしてください	

CTSU の下にぶら下がっているスタックに New - Transfer(r\_dtc)を追加

AL/Common Stacks	🛃 New Stack > 🛛 🟦 Extend Stack > 👔 Remov
g_ioport I/O Port (r_ioport)	de Touch (rm_touch)
(i)	0
	0
	追加後
	(ボックスの下のラインがピンクから グレーに変わります)

![](_page_19_Picture_8.jpeg)

![](_page_20_Picture_0.jpeg)

#### (5)UART の追加

The formation of the fo	ART Driver r of QE		
	New	> 🕀	UART (r_sci_uart)

#### Add SCI UART Driver...のボックスで「左クリック」New - UART(r\_sci\_uart)を追加。

HAL/Com	nmon Stacks	🔄 New Stack > 🔮 Extend Stack > 🙀 Remove	
т	ouch (rm_touch)		
1	•		
<b>₩</b> (	CTSU (r_ctsu)	l g_uart_qe UART (r_sci_uart)	
(1)		/	
(r (r () () ()	transfero Transfer _dtc) CTSU WRITE Write request tterrupt)	Add DTC Driver for Transmission [Recommended but optional]	※UART でも DTC 追加は可能です (サンプルプログラムでは追加はして いません)
	<u></u>		<u>}</u>
🔲 プロパテ	🗡 🗶 問題 🥘 スマート・ブラウザー		📑 8 🗖 🗖
g_uart_q	e UART (r_sci_uart)		
Settings	プロパティ ✓ Common	値	^
Artimo	Parameter Checking	Default (BSP)	
	FIFO Support	Disable	
	Elow Control Support	Disable	
	✓ Module g_uart_ge UART (r_sci_uart)		
	✓ General		
1	Name	🔒 g_uart_qe	
1	Channel	9	
	Data Bits	🛱 8bits	*

General - Channel:9に変更

![](_page_20_Picture_6.jpeg)

![](_page_21_Picture_0.jpeg)

# 2.3.3. Pins タブ

Pin Configuration						Generate Project Content
Select Pin Configuration		📑 Export to C	SV file 🔚 Confi	igure Pin Driv	er Warnings	
R7FA2L1AB2DFP.pincfg 🗸	Manage configurations	Gener	rate data: g_bsp	o_pin_cfg		
Pin Selection $\blacksquare \boxdot \Box \downarrow_z^a$	Pin Configuration					😲 Cycle Pin Group
Type filter text	Name	Value	Lock	Link		^
	Operation Mode	Disabled	¥			
Analog:ACIVIF A	✓ Input/Output	Disabled		$\langle \rangle$		
Analog:ADC	TSCAP	Enabled		$\Rightarrow$		
	TS00	None		$\Rightarrow$		
Connectivity:CAN	TS02-CFC	None		$\Rightarrow$		
Connectivity:IC	TS04	None		$\Rightarrow$		
Connectivity:SCI	TS05	None		$\Rightarrow$		
Connectivity:SPI	TS06	None		$\Rightarrow$		
v Input:CTSU	TS07	None		$\Rightarrow$		
CTSU0	TS08-CFC	None		$\Rightarrow$		
	TS09-CFC	None		$\Rightarrow$		
Input:KINT	TS10-CFC	None		$\Rightarrow$		
Monitoring:CAC	TS11-CFC	None		$\Rightarrow$		~
System:CGC	<					>
System:DEBUG     System:SYSTEM     Timers:AGT     Timers:GPT	Module name: CTSU0					
端子機能 端子番号						
Summary BSP Clocks Pins Interrupts Event Links	Stacks Components					

Input:CTSU - CTSU0

Operation Mode : Enabled に変更

ここで、タッチキーで使用する端子を設定します。

	割り当てる端子	S16A(自己容量基板)	D55A(相互容量基板)
TSCAP	P112		
TS13-CFC	P104	5	TX-0
TS14-CFC	P103(*1)	0	TX-1
TS15-CFC	P102(*2)	6	TX-2
TS16-CFC	P101	1	TX-3
TS26-CFC	P100	7	TX-4
TS28-CFC	P015	2	RX-0
TS30-CFC	P010	9	RX-4
TS31-CFC	P011	4	RX-3
TS32-CFC	P012	8	RX-2
TS33-CFC	P013	3	RX-1

上記の TSxx を Enabled に変更します。

※CTSU, TSxx は端子は複数からは選べません(有効にするか無効にするかの選択です)

(\*1)マイコンボード(HSBRA2L1F100)側で CAN のポートに割り当てられていますので、マイコンボード上の J14 のジャンパ2本を抜いて、P103, P102の端子を CAN の回路と切り離してください。

![](_page_22_Picture_0.jpeg)

Pin Configuration				😲 Cycle Pin Group
Name	Value	Lock	Link	1
Operation Mode	Enabled			
<ul> <li>Input/Output</li> </ul>				
TSCAP	✓ P112	l 💼		
TS00	None		$\Rightarrow$	
TS02-CFC	None		$\Rightarrow$	
TS04	None		$\Rightarrow$	
TS05	None		$\Rightarrow$	
TS06	None		$\Rightarrow$	
TS07	None		$\Rightarrow$	
TS08-CFC	None		$\Rightarrow$	
TS09-CFC	None		$\Rightarrow$	
TS10-CFC	None		$\Rightarrow$	
TS11-CFC	None		$\Rightarrow$	
TS12-CFC	None		$\Rightarrow$	
TS13-CFC	✓ P104	e filler and the second se	$\Rightarrow$	
TS14-CFC	✓ P103	E C	$\Rightarrow$	
TS15-CFC	✓ P102	- F	$\Rightarrow$	

設定後は、TSCAP, TSxx の端子の一部が有効化された状態となります。

Pin Configuration					Generate Project Content
Select Pin Configuration		Export to C	SV file 🔚 Confi	igure Pin Driver Warn	ings
R7FA2L1AB2DFP.pincfg	Manage configurations	🗹 Gener	rate data: g_bsp	o_pin_cfg	
Pin Selection $\blacksquare \ \boxdot \ \Box \ \downarrow^a_z$	Pin Configuration				😲 Cycle Pin Group
Type filter text         ✓       Peripherals         >       Analog:ACMP         >       Analog:ADC         >       ✓ Analog:ANALOG         >       Analog:ADC         >       Connectivity:CAN         >       Connectivity:SCI         SCI0       SCI1         SCI2       SCI3         SCI9       >         >       Connectivity:SPI         ✓       TSU0         >       Input:ICU	Name Pin Group Selection Operation Mode V Input/Output TXD9 RXD9 SCK9 CTS9 SDA9 SCL9 CL9 C Module name: SCI9	Value Value Mixed Disabled None None None None None None None			
Input:KINT     Monitoring:CAC     Sustem:CGC	Usage: When using Sim When switching	ole I2C mode, ensure po between I2C and other r	rt pins output typ modes, first disab	oe is n-ch open drain Ile.	
端子機能 端子番号 Summary BSP Clocks Pins Interrupts Event Links	s Stacks Components				

Connectivity:SCI - SCI9 の設定を変更します。

RA タッチキー評価キット[QEforCapTouch 編] 株式会社 **北非電子** 

![](_page_23_Picture_0.jpeg)

Pin Configuration					Generate Project Content	
Select Pin Configuration		Export to CSV file	e 🔚 Confi	igure Pin Driver Warnings		
R7FA2L1AB2DFP.pincfg 🗸 🗸	Manage configurations	🗹 Generate d	ata: g_bsp	o_pin_cfg		
Pin Selection $\exists \exists \exists \exists \exists \exists a \\ \exists z \end{bmatrix}$	Pin Configuration				Generate Project Cont	tent
Type filter text <ul> <li>Peripherals</li> <li>Analog:ACMP</li> <li>Analog:ADC</li> </ul>	Name Pin Group Selection Operation Mode V Input/Output	Value Mixed Asynchronous UART	Lock	Link	を押して設定をソースコー反映させる	-12
<ul> <li>&gt; ✓ Analog:ANALOG</li> <li>&gt; Analog:DAC</li> <li>&gt; Connectivity:CAN</li> <li>&gt; Connectivity:IIC</li> </ul>	TXD9 RXD9 SCK9 CTS9 SDA9	P109     P110     None     None     None				
SCI0 SCI1 SCI2 SCI3	SCL9	None		<b>+</b>		
	<				>	
> InputICU > InputKINT > Monitoring:CAC	Module name: SCI9 Usage: When using Simpl When switching b	le I2C mode, ensure port pin between I2C and other mode	s output typ s, first disab	pe is n-ch open drain. ole.		
端子機能 端子番号 Summary BSP Clocks Pins Interrupts Event Link	s Stacks Components					

Operation Mode Asyncronous : UART

TXD9 : P109 RXD9 : P110

を選択

ー通り設定が終わったら、「Generate Project Content」のボタンを押してソースコードを出力します。

![](_page_23_Picture_6.jpeg)

![](_page_24_Picture_0.jpeg)

# 2.4. エミュレータ(デバッガ)の接続設定

プロジェクトエクスプローラーから、対象プロジェクトを右クリック。

![](_page_24_Picture_3.jpeg)

#### デバッグ – デバッグの構成

![](_page_24_Picture_5.jpeg)

![](_page_24_Picture_6.jpeg)

![](_page_25_Picture_0.jpeg)

(1)使用エミュレータ

使用するエミュレータを選択してください。

※E2 をマイコンボード 14P コネクタで使用する場合は J8 ジャンパを 1-2 側に設定。E2Lite を使用する場合は、2-3 側に設定してください。(オプションの 20P コネクタを使用する場合はジャンパの設定は不要です。)

Connection Settings - エミュレータから電源を供給する: いいえ

を選んでください。

「適用」、「閉じる」で設定画面を閉じてください。

※エミュレータから電源を供給する事も可能ですが、その場合は ・<u>別な箇所(電源コネクタ等)から電源を供給しない</u> 様にしてください。

※なお、E2Lite をお使いの場合は、エミュレータから供給する電圧は、3.3V しか選択できません(E2 では、5V と 3.3V が選択可能です)

※「エミュレータ」と「デバッガ」、「Debugger」は同義です

![](_page_25_Picture_10.jpeg)

![](_page_26_Picture_0.jpeg)

# 3. QE CapTouch の使用

# 3.1. CapTouch の起動

e2studio + FSP 環境でプロジェクトを作成し、

![](_page_26_Picture_4.jpeg)

Renesas Views - CapTouch メインを実行します。

![](_page_26_Picture_6.jpeg)

全体の流れ(ワークフロー)が表示されます。

(1)プロジェクトの選択

新しく作成したプロジェクトを指定します。

![](_page_26_Picture_10.jpeg)

![](_page_27_Picture_0.jpeg)

(2)構成の選択

<b>構成の選択</b> タッチインタフェース構成を選択/作成 します。	
Ý	
タッチインタフェース構成の新規作成	

タッチインタフェース構成の新規作成

を選びます。

# 3.2. タッチキーインタフェースの作成

#### 3.2.1. 自己容量基板(S16A)

タッチインタフェース構成の作成			
ッチインタフェース構成のファイル	۲۵:         RA2L1_QE_CAP_TOUCH_S	構成(メソッド)の設定	構成の流用/再編集
明:			
			タッチI/F
			·····································
Butt	n00		目亡帝軍
			ボタン
			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
			診断コード用端子
定			タッチI/Fの削除
タッチI/Fの設定	総抵抗の設定割り付けTSxの解除		
設定内容に問題があります			
		作成	:( <u>C</u> ) キャンセル ヘルプ( <u>H</u>

「ボタン」を押し、ボタンをレイアウト上に配置していきます。

![](_page_28_Picture_0.jpeg)

	成のファイル名:	RA2L1_QE_CA	P_TOUCH_S	構成(メソッド)の設定		構成の流用/再編集
	Button00 I	Button01 Button02	Button03 Button04			タッチルF           静電容量方式           自己容量
		Button05 Button06	Button07 Button08	Button09		ボタン スライダ (横方向)
						スライダ(縦方向)
						ホイール
						キーパッド
	_				ーパッドは 👦	3Dジェスチャ (Al)
	•	S16А нокито	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	RA のタッ 使えませ	ッチキーキットでは <sup>●</sup> ADE IN JAPAN	タッチパッド
	TS01	TS03 TS05	5 TS06	TS08 TS10 TS13	TS14 TS15	シールド端子
	0	1 2	3	4 📈 🗗	8	温度補正端子
						容量センサ
					TSII	電流センサ
	•	5 0			. 14	診断コード用端子
	S16A :	基板のキーパ	ッド 0~9 を	·		タッチI/Fの削除
タッチI/Fの	画面上	に配置する				

RA2L1 のタッチキーキットでは、0~9 の 10 個のキーパッドが使用可能なので、同じような画面イメージとなる様に 10 個のボタンを画面上に配置します。

![](_page_28_Figure_3.jpeg)

Button00をダブルクリックして、このボタンに関する設定を行います。

接続されている TS 端子は、TS14 なので、TS14 を選択します。抵抗値は、560 Ω(デフォルト値)のままとします。 (S16A 基板上に、各キーパッドに対し、560 Ωの抵抗が搭載されています)

また、名前を Button00 から Button0 に変更します。(名前は任意です。数字で始まる名前やスペースは許されていませ。)

![](_page_28_Picture_7.jpeg)

RA タッチキー評価キット[QEforCapTouch 編]

![](_page_29_Picture_0.jpeg)

#### ・キーパッドと TS の対応

S16A(自己容量基板) キーパッド	TSxx 番号
0	TS14
1	TS16
2	TS28
3	TS33
4	TS31
5	TS13
6	TS15
7	TS26
8	TS32
9	TS30

#### 上記表に従い、キーパッドに TSxx の割り当てを行います。

![](_page_29_Figure_4.jpeg)

作成を押す。

![](_page_29_Picture_6.jpeg)

![](_page_30_Picture_0.jpeg)

#### 3.2.2. 相互容量基板(D55A)

#### 相互容量基板を用いる際のタッチインタフェースの作成に関しては、以下の様になります。

/f1/y97I-ス構成のファイル4: RA2L1_QE_CAP_TOUCH_M 構成(メソyト)の設定 明: Keypad00 赤色の部分をダブルクリック して設定を開く	構成の流用/再編9 タッチI/F
k Keypad00 赤色の部分をダブルクリック して設定を開く	タッチI/F
Keypadco 赤色の部分をダブルクリック して設定を開く	タッチI/F
Keypad00 赤色の部分をダブルクリック して設定を開く	
Keypad00 赤色の部分をダブルクリック して設定を開く	一日本 日本 日
Keypadoo 赤色の部分をダブルクリック して設定を開く	HILLE
赤色の部分をダブルクリック して設定を開く	ボタン
赤色の部分をダブルクリック して設定を開く	スライダ(横方向)
赤色の部分をダブルクリック して設定を開く	スライダ(縦方向)
赤色の部分をダブルクリック して設定を開く	ホイール
赤色の部分をダブルクリック して設定を開く	キーパッド
赤色の部分をダブルクリック して設定を開く	3Dジェスチャ (Al)
して設定を開く	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
	診断コード用端子
	タッチI/Fの削除
タッチ//Fの設定         総抵抗の設定         割り付けTSxの解除	
設定内容に問題があります。	

![](_page_30_Figure_4.jpeg)

#### 静電容量方式: 相互容量

![](_page_31_Picture_0.jpeg)

送信用のところに、TS13~TS26を設定。受信用に、TS28~TS30を設定する。抵抗値は、560Ω(デフォルトのまま) としてください。

「キーパッドボタン(相互)の設定」を押す。

キーパッドボ	タン(相互)					キーパッドボ	タン(相互)				
名前						—— 名前					
	TS13	TS14	TS15	TS16	TS26		TS13	TS14	TS15	TS16	TS26
TS28	Koo_Boo	K00_B01	K00_B02	K00_B03	K00_B04	TS28	КО	K1	K2	КЗ	K4
TS33	K00_B05	K00_B06	K00_B07	K00_B08	K00_B09	TS33	K5	K6	K7	K8	K9
TS32	K00_B10	K00_B11	K00_B12	K00_B13	K00_B14	TS32	K10	K11	K12	K13	K14
TS31	K00_B15	K00_B16	K00_B17	K00_B18	K00_B19	TS31	K15	K16	K17	K18	K19
TS30	K00_B20	K00_B21	K00_B22	K00_B23	K00_B24	TS30	K20	K21	K22	K23	K24

キーの名前を、K0~K24(基板に記載されている 0~24 に対応)に変更。

※数字から始まる名前に設定できないので Kn としています

チインタフェース構成のファイル名:	[	RA2L1_QE_	CAP_TOUCH	_M		構成(メソッド)の設定		構成の流用
								ータッチI/F
				Keypad00			•	
	TS28	КО	K1	K2	Кз	K4	IIS00-05	TS04-05
	1233	К5	K6	K7	К8	К9	01-06 55A	9 9 33-06 8 8
	T\$32	K10	K11	K12	K13	K14	1500- 10	1504- 1503- 1502- 1702-
	TS31	K15	K16	K17	K18	K19	1501−08 1501−08	1503-08 18 18 18 17 17
	TS30	K20	K21	K22	K23	K24		
		TS13	TS14	TS15	TS16	TS26		4-09 3-09 2-09
							D55A 基板とは 90 J	<b>変傾いた配置とな</b> り
							0~24 のキーパッドた	<sup>バ</sup> K0~K24 に対応
							となります	
を タッチI/Fの設定	総抵抗	の設定	生い	付けTSxの解	Ŷ			タッチI/F
7777710782.AL	1015176	V BX.AL		1310 134 00 (311)	25			

![](_page_31_Picture_7.jpeg)

![](_page_32_Picture_0.jpeg)

# 3.3. タッチキーインタフェースの調整

マイコンボードとタッチキーボード基板(S16A または D55A)を接続。

#### (3)調整

2. 調整	E
各タッチセンサについっ テいます。	(自動調整処理を
<u>ターゲットボードの</u> エミュレータ経由でタ PCを接続します。	<b>D接続</b> マーゲットボードと
調整の開始 ダイアログの指示に従	い操作します。
調整を開	始する
宮度な設定	を方がにする

「調整を開始する」を押す。

プログラムのビルドとデバッガへのダウンロードが実行されます。

![](_page_32_Picture_7.jpeg)

上記メッセージが出た場合は、(どちらでも構いませんが)ここでは「いいえ」を押します。

![](_page_32_Picture_9.jpeg)

上記のメッセージが出力され、1/15, 2/15, …の様に処理が進んでいきます。

![](_page_32_Picture_11.jpeg)

RA タッチキー評価キット[QEforCapTouch 編]

![](_page_33_Picture_0.jpeg)

#### 3.3.1. 自己容量基板(S16A)

![](_page_33_Picture_2.jpeg)

キーパッドにタッチする事が求められますので、表示に従い S16A ボードの 5 のキーパッドにタッチして、(PC の)キ ーボードを叩いてください。

順次他のキーパッドに関しても、同様にタッチ&キーボードを叩くという事を繰り返してください。

※キーパッドは TS 順の入力となりますので、5→0→…の様に 0 からの順番ではない事に注意願います

(2) 自動調整処理中         ×										
感度調整で警告 沢してリトライ( 理を継続してくだ	きやオーバ: 再調整) ざい。	-フロ- してく:	等のエラ ださい。 『	ーが検出さ 問題がなけ	れてい れば、	\る場合、対 「調整処理	す象とするタ 『の継続」ボ	ッチセンサを選 タンを押して処		
リトライ対象の選択	メソッド	種別	名前	タッチセンサ	閾値	オーバーフロー	警告/I5-			
	config01	ボタン	Button0	TS14	1309					
	config01	ボタン	Button1	TS16	1026					
	config01	ボタン	Button2	TS28	1043					
	config01	ボタン	Button3	T\$33	1105					
	config01	ボタン	Button4	TS31	1051					
	config01	ボタン	Button5	TS13	1266					
	config01	ボタン	Button6	TS15	16					
	config01	ボタン	Button7	TS26	1197					
	config01	ボタン	Button8	TS32	1301					
	config01	ボタン	Button9	TS30	1242					
リトライ 調整処理	の継続									
							キャンセル	ヘルプ( <u>H</u> )		

上記の様に「閾値」の値が極端に小さいボタンがあった場合タッチに失敗しています。「リトライ対象の選択」にチェッ クを入れ、「リトライ」を行ってください。

※閾値が 65535 になっている場合も NG です(多分計算結果が負数となり表示上、65535 になっていると思われます)

値が問題なさそうであれば、「調整処理の継続」を押して次に進んでください。

![](_page_33_Picture_10.jpeg)

![](_page_34_Picture_0.jpeg)

#### 3.3.2. 相互容量基板(D55A)

![](_page_34_Picture_2.jpeg)

キーパッドにタッチする事が求められますので、表示に従い D55A ボードの 0 のキーパッドにタッチして、(PC の) キーボードを叩いてください。

順次他のキーパッドに関しても、同様にタッチ&キーボードを叩くという事を繰り返してください。

※キーの順番ですが、 $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 20 \rightarrow \cdots \geq 5$ 個のキー(列方向)は順番ですが、行はばらばらの順となる事に注意ください

0	自動調整処理中								×			
感度 択し 理を	感度調整で警告やオーバーフロー等のエラーが検出されている場合、対象とするタッチセンサを選択してリトライ(再調整)してください。問題がなければ、「調整処理の継続」ボタンを押して処理を継続してください。											
IJ	トライ対象の選択	メソッド	種別	名前	タッチセンサ	閾値	オーバーフロー	警告/I5-				
	]	config01	キーパッド ボタン	КО	[TX]TS13, [RX]TS28	1006						
	]	config01	キーパッド ボタン	K1	[TX]TS14, [RX]TS28	1238						
	]	config01	キーパッド ボタン	K2	[TX]TS15, [RX]TS28	958						
	]	config01	キーパッド ボタン	КЗ	[TX]TS16, [RX]TS28	993						
	]	config01	キーパッド ボタン	K4	[TX]TS26, [RX]TS28	1131						
	]	config01	キーパッド ボタン	K5	[TX]TS13, [RX]TS33	973						
	]	config01	キーパッド ボタン	K6	[TX]TS14, [RX]TS33	1053						
	]	config01	キーパッド ボタン	K7	[TX]TS15, [RX]TS33	1099						
	]	config01	キーパッド ボタン	K8	[TX]TS16, [RX]TS33	1123						
	]	config01	キーパッド ボタン	K9	[TX]TS26, [RX]TS33	1207						
	]	config01	キーパッド ボタン	K10	[TX]TS13, [RX]TS32	1021						
	]	config01	キーパッド ボタン	K11	[TX]TS14, [RX]TS32	1005						
	]	config01	キーパッド ボタン	K12	[TX]TS15, [RX]TS32	1019						
	]	config01	キーパッド ボタン	K13	[TX]TS16, [RX]TS32	1084						
	]	config01	キーパッド ボタン	K14	[TX]TS26, [RX]TS32	1160						
	]	config01	キーパッド ボタン	K15	[TX]TS13, [RX]TS31	997						
	]	config01	キーパッド ボタン	K16	[TX]TS14, [RX]TS31	1035						
	]	config01	キーパッド ボタン	K17	[TX]TS15, [RX]TS31	1042						
	]	config01	キーパッド ボタン	K18	[TX]TS16, [RX]TS31	1147						
	]	config01	キーパッド ボタン	K19	[TX]TS26, [RX]TS31	1625						
	]	config01	キーパッド ボタン	K20	[TX]TS13, [RX]TS30	1044						
	]	config01	キーパッド ボタン	K21	[TX]TS14, [RX]TS30	998						
	]	config01	キーパッド ボタン	K22	[TX]TS15, [RX]TS30	996						
	]	config01	キーパッド ボタン	K23	[TX]TS16, [RX]TS30	1092						
	]	config01	キーパッド ボタン	K24	[TX]TS26, [RX]TS30	1311						
IJł	ライ調整処理	の継続					キャンセル	ヘルプ( <u>H</u> )				

閾値が他より極端に小さい、65535 になっている、オーバフローやエラーがない場合は「調整処理の継続」で次に進

む。

![](_page_34_Picture_9.jpeg)

![](_page_35_Picture_0.jpeg)

# 3.4. ソースコードの変更

(4)パラメータファイルの出力

![](_page_35_Picture_3.jpeg)

「ファイルを出力する」。

#### (5)ユーザプログラム内にタッチキーの判定プログラムを取り込む

![](_page_35_Picture_6.jpeg)

#### 「例を表示する」。

📴 サンプルコードの表示		×						
main()関数のコード例:								
¥	*****	^						
* FILE : qe_sample_main.c * DATE : 2021-06-01								
* DESCRIPTION : Main Program for F *	A							
* NOTE:THIS IS A TYPICAL EXAMPLE *								
*****	***************************************							
#include "qe_touch_config.h" #define TOUCH_SCAN_INTERVAL_EXAMPLE (20) /* milliseconds */								
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								
2		¥						
	7-7464							
シリッフルートにコピー	ノアイルに田刀	アノリソーションノートの表示						
		OK ヘルプ( <u>H</u> )						

ユーザプログラム内に記載するテンプレートが出力されるので、この記述を参考に、 src/hal\_entry.c (ユーザプログラムの main 関数に相当する) に、タッチキーのプログラムコードを追加してください。

![](_page_35_Picture_10.jpeg)

![](_page_36_Picture_0.jpeg)

ここでは、ファイルに出力を押して、QE Cap Touch が生成したプログラムをファイルに保存します。

![](_page_36_Picture_2.jpeg)

上記に保存されます。

サンプルプロジェクト、

RA2L1\_QE\_CAP\_TOUCH\_S(自己容量)

RA2L1\_QE\_CAP\_TOUCH\_M(相互容量)

には、LCDを使った表示例が、hal\_entry.c内に記載されています。

![](_page_36_Picture_8.jpeg)

上記ファイルが、一般的な main 関数に相当する hal\_entry()(FSP が呼び出すユーザプログラムのメイン関数)を 含むファイルですので、このファイルにユーザプログラムを記載、または hal\_entry()から別ファイルに記載したプログ ラムを呼び出す様にしてください。

![](_page_36_Picture_10.jpeg)

![](_page_37_Picture_0.jpeg)

## 3.4.1. 自己容量基板(S16A)

src/hal\_entry.c

![](_page_37_Figure_3.jpeg)

#### <mark>黄色部分</mark>を追加。

•qe\_gen/qe\_sample.c

![](_page_37_Picture_7.jpeg)

![](_page_38_Picture_0.jpeg)

・qe\_gen/qe\_sample.c(続き)

```
void qe_touch_main(void)
{
   fsp_err_t err;
   int i;
   unsigned long x;
   <mark>unsigned char</mark> key_table[10] = {5, 0, 6, 1, 7, 2, 9, 4, 8, 3};  //測定<u>ch</u>とキーパッド順のテーブル
   char result[10];
                                          測定 ch の若い順-キーパッドの対応
                                          のテーブルを定義
   lcd_hs1();
   11
                1234567890123456
   lcd_write_str("0123456789ABCDEF");
   /* 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) {}
   }
   /* Main loop */
   while (true)
   {
       /* 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, NULL, NULL);
       if (FSP_SUCCESS == err)
       {
                                                          LCD 画面に、タッチしているキーの情報を表示
                                                          (タッチ:o, タッチしていない:-)
          /* TODO: Add your own code here. */
          lcd_hs2();
                                                          表示例一
          x = 0 \times 1 U L;
          for(i=0; i<10; i++)</pre>
                                                          0123456789ABCDEF
          {
                                                          --0---0---
              if((x & button_status) == 0)
              result[key_table[i]] = '-'; //-(not touch)
}___
              else
              {
                 result[key_table[i]] = 'o'; //o(touch)
             }
              x <<= 1UL;
          //結果のLCD表示
          lcd write str(result);
       }
 (後略)
```

![](_page_38_Picture_4.jpeg)

![](_page_39_Picture_0.jpeg)

### 3.4.2. 相互容量基板(D55A)

src/hal\_entry.c

![](_page_39_Figure_3.jpeg)

#### <mark>黄色部分</mark>を追加。

· ge gen/ge sample.c

```
* FILE : ge sample main.c
* DATE : 2021-06-01
* DESCRIPTION : Main Program for RA
* NOTE: THIS IS A TYPICAL EXAMPLE.
#include "qe_touch_config.h"
#include "../src/lcd_1602/lcd_1602.h"
(中略)
```

![](_page_39_Picture_7.jpeg)

![](_page_40_Picture_0.jpeg)

・ge gen/ge sample.c(続き)

```
void qe_touch_main(void)
{
   fsp_err_t err;
   int i;
   unsigned long x;
   unsigned char key_table[25] = {0, 1, 2, 3, 4, 20, 21, 22, 23, 24, 15, 16, 17, 18, 19,
                <mark>10,11,12,13,14,5,6,7,8,9}; //測定<u>ch</u>とキーパッド順のテーブル</mark>
   int result;
                                                      測定 ch の若い順-キーパッドの対応
   lcd hs1();
                                                      のテーブルを定義
                1234567890123456
   11
   lcd_write_str("Touch key pad: ");
   /* 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) {}
   }
   /* Main loop */
   while (true)
   {
       /* 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, NULL, NULL);
       if (FSP SUCCESS == err)
       {
          /* TODO: Add your own code here. */
          lcd hs2();
          lcd write str("0x");
          lcd_write_uint32_hex((unsigned long)button_status);
          lcd_write_str(" > ");
          x = 0x1UL;
          result = -1;
                                                 LCD 画面に、タッチしているキーの情報を表示
          for(i=0; i<25; i++)</pre>
                                                 表示例-
              if(button status & x)
              {
                                                Touch Key pad:
0x00000010 > 4
                 result = i;
                 break;
              }
                                                      1
              x <<= 1UL;
                                                                   タッチキーパッド番号
                                                 button status
          if(result == -1)
                                                 button status は、64bit 変数で、下位 32bit を 16
                                                 進数で表示
              lcd write str("- ");
          }
          else
          {
              lcd_write_uint8(key_table[i]);
 (後略)
```

![](_page_41_Picture_0.jpeg)

# 3.5. ビルド・実行

workspace_RA - RA2L1_QE_CAP_TOUCH_S/qe_c     環集(E) ソース(S) リファクタリング(T) ナ	gen/qe_touch_sa ビゲート( <u>N</u> ) 検卵	ample.c 索( <u>A</u> )	: - e² st プロジェ	udio クト( <u>P</u> )	Renesas <u>V</u>
<ul><li>参 ■ 参 デバッグ(B) ~</li></ul>	RA2L1_QE	CAP_T	OUCH_	S Debu	ıg_f 🗸 🌼
🔁 プロジェクト・エクスプローラー 🔀	E 🕏	7	8 🗆		🖒 CapTou
V 😂 RA2L1_QE_CAP_TOUCH_S [Debug]				^	22
> ぷ バイナリー					23
> 🔊 Includes					24
✓ P ge gen					25
> c ae touch confia.c					26
B ge touch config.h					27
					28

#### 「ビルド」を実行。

![](_page_41_Picture_4.jpeg)

エラーや意図しないワーニングがなければ問題ありません。

・エミュレータ接続(E2, E2Lite 等をお持ちの場合)

・マイコンにプログラムを書き込む(デバッガをお持ちでない場合)

(ハードウェア編のマニュアルを参照し、製品付属の USB-ADAPTER-RX14 でプログラムの書き込みを行ってください)

どちらかの方法で作成したプログラムをRA2L1のマイコンに書き込みを行ってください。

以下は、エミュレータ接続のケースで説明します。

![](_page_41_Picture_11.jpeg)

![](_page_42_Picture_0.jpeg)

#### ※E2 エミュレータを使用する場合

😰 デバッグ構成		_	
構成の作成、管理、および実行			Ť
☐ @ ŵ @ ¥	名前(N): RA2L1_QE_CAP_TOUCH_S Debug_Flat		
<ul> <li>E C/C++ アプリケーション</li> <li>E C/C++ リモート・アプリケーション</li> <li>■ EASE Script</li> </ul>	Debug hardware: E2 (ARM) V Target Device: R7FA2L1A8		
GDB OpenOCD Debugging	GDB Settings Connection Settings デバッグ・ツール設定		
GDB Simulator Debugging (RH850)	✓ クロック		^
C GDB ハードウェア・デバッギング	メイン・クロック・ソース	内部クロック	~
Java アブリケーション	外部クロック入力周波数 (MHz)		
🜌 Java アフレット	内蔵フラッシュ・メモリー書き換え時にクロック・ソースの変更を許可する	はい	~
✓ c <sup>™</sup> Renesas GDB Hardware Debugging	動作周波数 (MHz)		
RA2L1_QE_CAP_TOUCH_S Debug_Flat	✓ ターゲット・ボードとの接続		
c ™ Renesas Simulator Debugging (RX, RL78)	エミュレーター	(Auto)	
目、リモート Java アフリケーション		SWD	¥
🛛 🖥 起動クループ	接続速度 (kHz)	SWD	
	✓ 電源	JTAG	
	エミュレーターから電源を供給する (MAX 200mA)	いいえ	¥
	電源供給先	ユーザインタフェース	$\sim$
	供給電圧 (V)	3.3	~
	✓ 接続		
	接続時にリセット状態を維持する	はい	~
	IDコード (バイト単位)	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	
	低消費電力モードを使用する	はい	~
	✓ TrustZone		
		11112	•
15 項目のうち 13 項目がフィルターに一致		前回保管した状態に戻す(V) 適 デパッグ(D)	5用(Y) 問じる
		77777-7	

#### タイプとして「SWD」を選択してください(デフォルト)。

![](_page_42_Picture_4.jpeg)

#### 「デバッグ」を実行。

workspace_RA - RA2L1_QE_CAP_TOUCH_S/ra/fsp/src/bsp/cmsis/Device/RENESAS/	Source/startup.c - e² studio			
ファイル(E) 編集(E) ソース(S) リファクタリング(T) ナピゲート(N) 検索(A) プロジェクト(P)	Renesas <u>V</u> iews 実行( <u>R</u> )	ウィンドウ( <u>W)</u> ヘルプ( <u>H</u> )		
	ug_f 🗸 🔅 🗄 🛨 🔚	💼   🛞 🕶 🔦 🕶 🔜 ! 🚍 ! 🏪 !	×   🕨 🕺 🖉 🛪	🔁 .r.   it 🗟 🛒 !
🏘 テバッグ 🛛 📄 🎋 🖇 🖳 🗖	ؼ CapTouchメイン (QE)	c main.c c hal_entry.c	🖻 qe_touch_sample.c	🖸 startup.c 🗙
✓ ☑ RA2L1_QE_CAP_TOUCH_S Debug_Flat [Renesas GDB Hardware Debugging]	⇒ 64 00004cd0	SystemInit();		
✓ A RA2L1_QE_CAP_TOUCH_S.elf [1] [cores: 0]	65			
🗸 🕐 Thread #1 1 (single core) [core: 0] (Suspended : シグナル : SIGTRAP:Trace/br	66 67 00004 add	/* Call user application.	*/	
Reset_Handler() at startup.c:64 0x4cd0	67 00004C00	main();		
📓 arm-none-eabi-gdb (7.8.2)	69 ⊖	while (1)		

ブレークポイントで止まりますので、「再開」ボタン(もしくは F8)で進めてください(2 回押す)。

![](_page_42_Picture_8.jpeg)

![](_page_42_Picture_9.jpeg)

![](_page_43_Picture_0.jpeg)

# 3.6. モニタリング

![](_page_43_Picture_2.jpeg)

CapTouch モニタのビューを有効化。

#### (6)シリアル接続

4. 動作確認	
タッチインタフェースの動作確認と微調整 を行えます。	モニタリングは、「エミュレータ接続」
<b>デバッグの開始(エミュレータ接続)</b> 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。	の2種類の接続方法があります。 エミュレータをお持ちでない場合も、「シリアル接続」は実行できます。
<b>シリアル接続</b> 王ミュレータを使わない場合、シリアル 通信によるモニタリング機能を有効にし ます。 ポーレート 115200 ポート番号 COM10 ↓ 接続する	
モニタリングの開始 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。 ピューを開く	

ポート番号を、USB-ADAPTER-RX14の接続ポートの番号を選択して「接続する」。

![](_page_43_Figure_7.jpeg)

「モニタリングを有効にする」。

![](_page_43_Picture_9.jpeg)

![](_page_44_Picture_0.jpeg)

#### ー自己容量タイプキーパッド(S16A)の場合-

![](_page_44_Figure_2.jpeg)

5 のタッチパッドにタッチした場合

・指のアイコンが出現

・カウント値が基準値+閾値を超えているとき、赤字になりタッチ判定される

となります。

#### (7)ビューを開く

😓 CapTouchメイン (QE)	🏷 CapTouchステータス・チャ	- ト (QE) 🗙 🔂 main.c	i hal_entry.c	c qe_touch_sample.c	🖻 startup.c	- 8
タッチI/F:	~ □ 選	択状態を同期する				
タッチ位置:	基準値:	閾値:	差分值:			
データ収集の開始						
ノイズ値[NT]:	平均值[NT]:	最小值:	最大値:			
ノイズ値[T]:	平均值[T]:	シグナル値:	SNR值:			
65535						
49149						
32766						
16202						
10303						
0						

![](_page_44_Picture_9.jpeg)

![](_page_45_Picture_0.jpeg)

CapTouch ステータスチャートが開きますので、タッチ I/F を観測したいキーパッド(ここでは、キーパッド 0;button0 とします)を選択。

发 CapTouchメイン (QE)	🏷 CapTouchステータス・チャ	-ト(QE) 🗙 💽 main.c	c hal_entry.c	c qe_touch_sample.c	.c startup.c	
タッチI/F: Button0@confi	g01 🗸 🗌 遵	択状態を同期する				
/F種別: ボタン(自己), チャネ)	l↓: TS14					
カウント値: 15416	5 基準値: 15409	) 閾値: 130	9 差分值:	7		
データ収集の開始						
ノイズ値[NT]:	平均值[NT]:	最小值:	最大値:			
ノイズ値[T]:	平均值[T]:	シグナル値:	SNR值:			
18906						
	調点	≥値				
18022	A [ X]					
						T
						<mark>タッチ判定</mark>
17139						
						閾値
16256						 
						非タッチ判定
15373				·		 其淮值

この期間タッチ

横軸時間、縦軸測定値 青線:基準値(非タッチ時の測定値) 赤線:測定値

緑線:閾値

上記の他にも

![](_page_45_Picture_7.jpeg)

観測できる項目がありますので、見てみてください。

![](_page_45_Picture_9.jpeg)

![](_page_46_Picture_0.jpeg)

2 3 3 B 3

> > •

.

ー相互容量タイプキーパッド(D55A)の場合-

![](_page_46_Figure_2.jpeg)

![](_page_46_Figure_3.jpeg)

QE CapTouch では、数値の変化がリアルタイム、グラフィカルに表示されますので、数値の変化が掴みやすいかと 思います。

![](_page_46_Picture_5.jpeg)

![](_page_47_Picture_0.jpeg)

![](_page_47_Picture_1.jpeg)

QE CapTouch を使用すると、タッチキーインタフェース(キーパッドに合わせたレイアウト)の作成を GUI で行う事 ができ、ソースコードの生成をツールが行ってくれますので、タッチキー(CTSU2)のレジスタ構成や、仕組みを理解し ていなくても、タッチキーを使用したアプリケーションの構築が可能です。

![](_page_47_Picture_3.jpeg)

![](_page_48_Picture_0.jpeg)

# 5. 付録

## 取扱説明書改定記録

バージョン	発行日	ページ	改定内容
REV.1.0.0.0	2022.2.9		初版発行

お問合せ窓口 最新情報については弊社ホームページをご活用ください。 ご不明点は弊社サポート窓口までお問合せください。

<sub>株式会社</sub> 北丰電子

〒060-0042 札幌市中央区大通西 16 丁目 3 番地 7 TEL 011-640-8800 FAX 011-640-8801 e-mail:support@hokutodenshi.co.jp (サポート用)、order@hokutodenshi.co.jp (ご注文用) URL:http://www.hokutodenshi.co.jp

商標等の表記について

・ 全ての商標及び登録商標はそれぞれの所有者に帰属します。

・ パーソナルコンピュータを PC と称します。

RA タッチキー評価キット[QEforCapTouch 編]

![](_page_48_Picture_11.jpeg)

ルネサス エレクトロニクス RA グループマイコン搭載 HSB シリーズマイコンボード向けキット

# RA2L1 タッチキー評価キット [QE CapTouch 編] マニュアル

<sub>株式会社</sub>

©2022 北斗電子 Printed in Japan 2022 年 2 月 9 日改訂 REV.1.0.0.0 (220209)