スタート作業ガイド（NS-RX231編）

e2studioベース

目次

1 プロジェクト生成

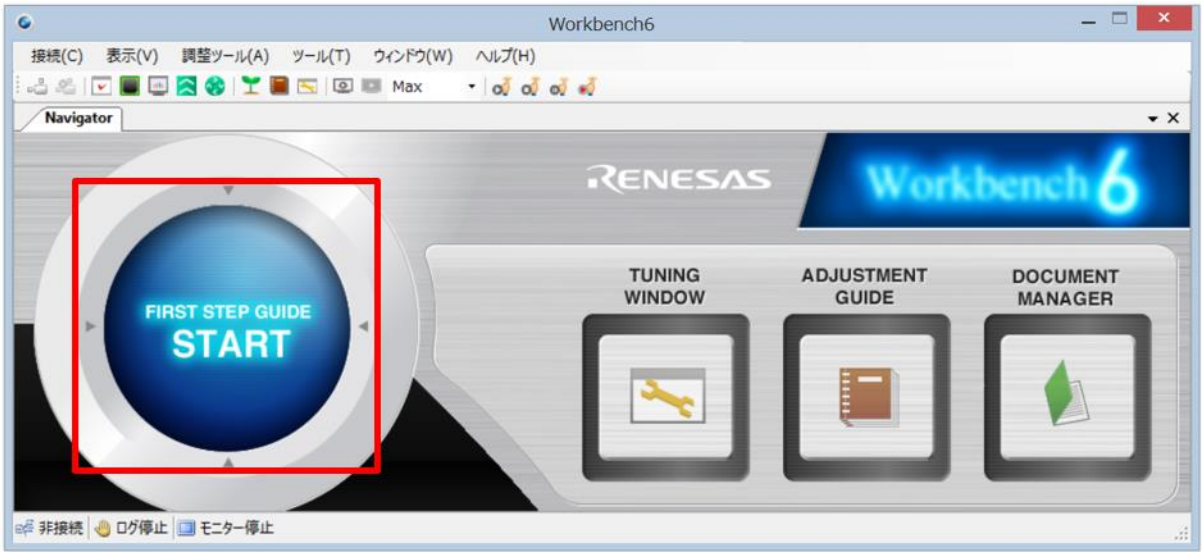
CTSU(Capacitive Touch Sensor Unit) APIを利用して簡単にプロジェクトを作ることができるWorkbench6のFirst Step Guide機能でソースを作ってみます。

本文を読み進めるのに先立って、Workbench6の機能を扱った動画を参照してください。

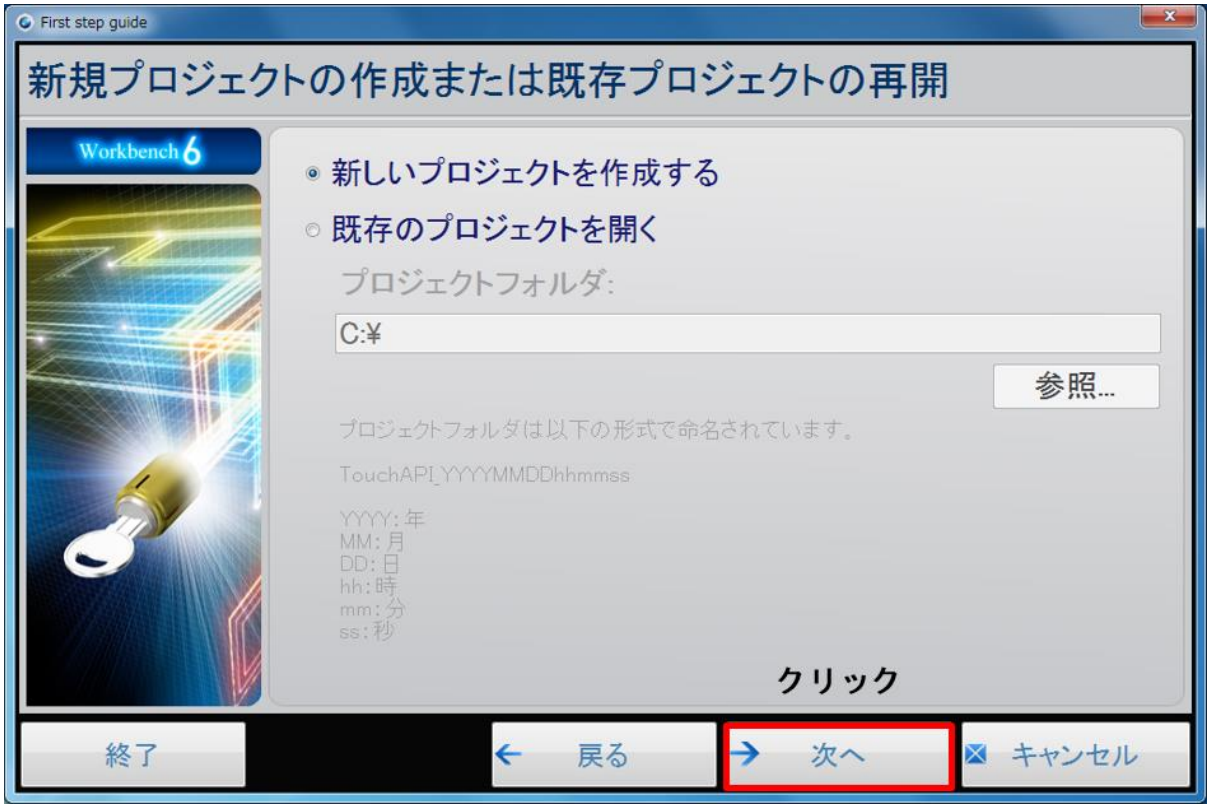
Integrated Development Environment for Renesas Capacitive Touch Workbench6(ver1.03)
https://www.youtube.com/watch?v=8oq5iyGyaMw①

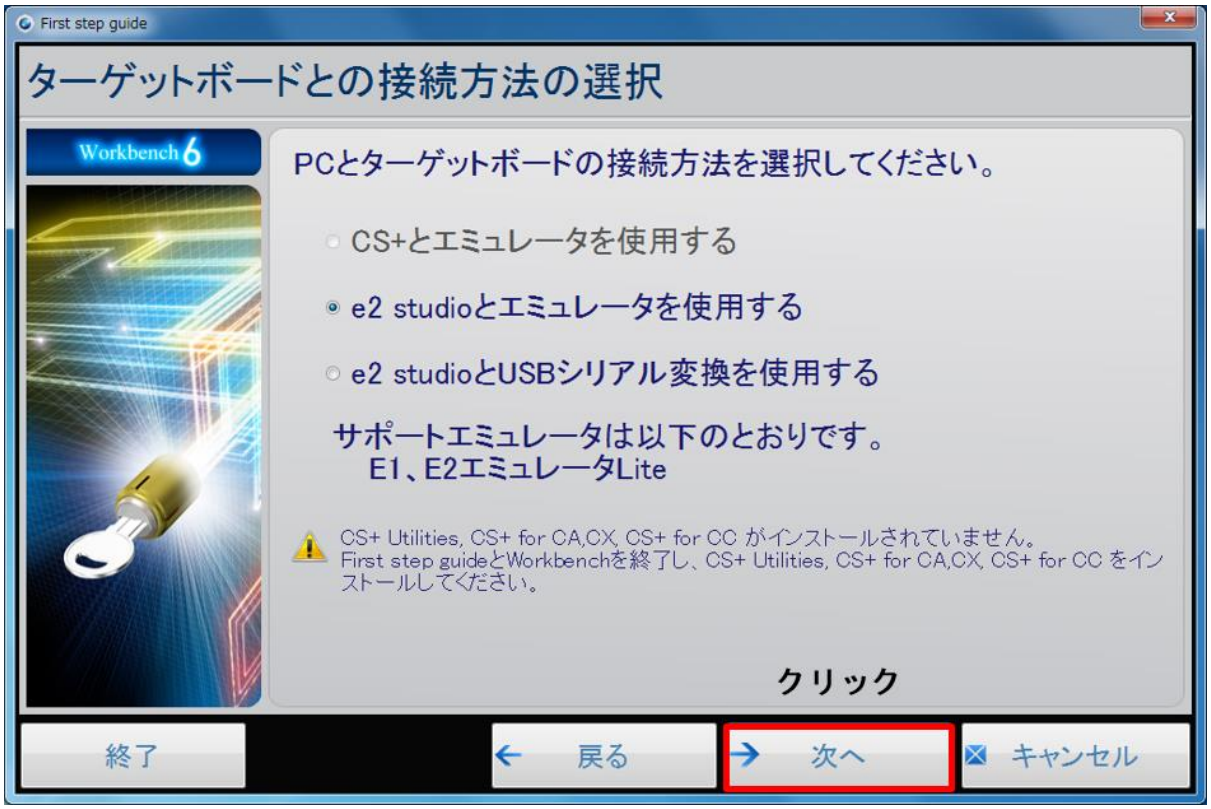
https://www.youtube.com/watch?v=tdIpiyTrKVM②

https://www.youtube.com/watch?v=2nQ9OcP5Cgg③

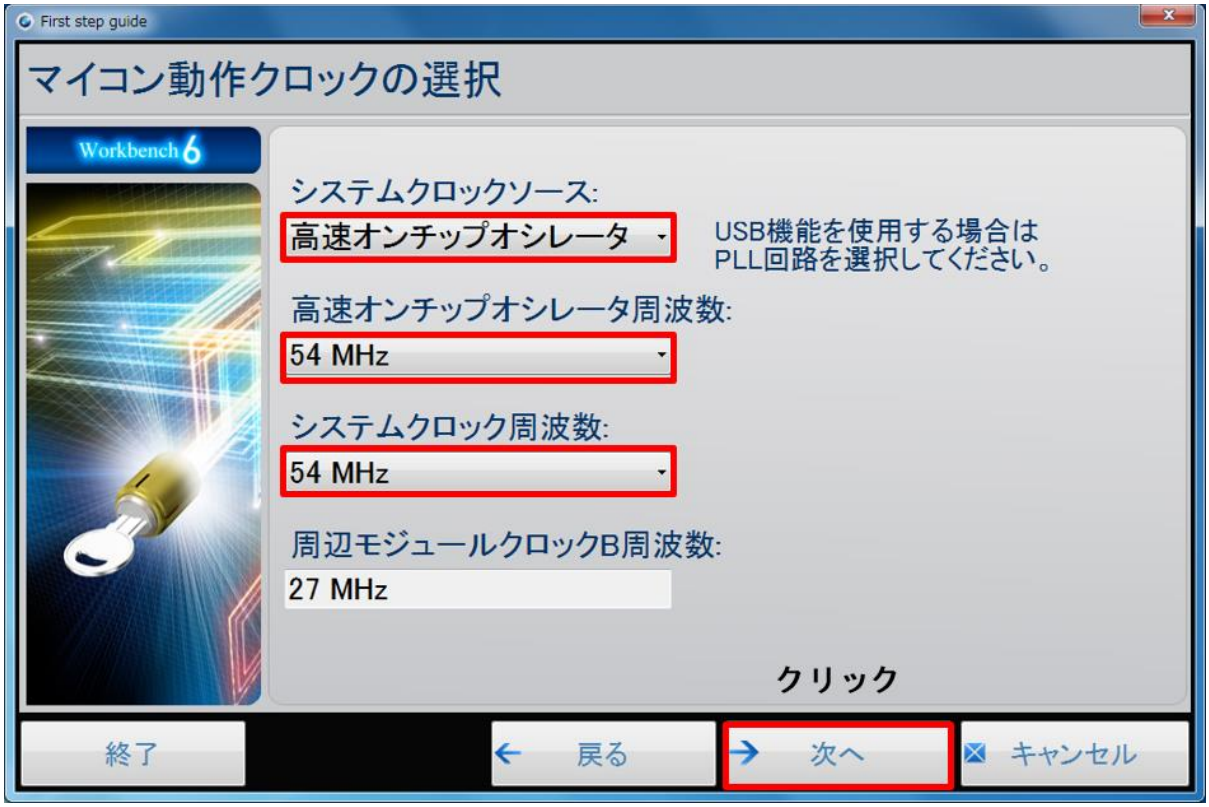



First step guide
RENESAS
ルネサス静電容量タッチソリューションにようこそ
Workbench 6
このウィザードではターゲットシステムに合わせて
適切に調整したファームウェアを生成します。
クリック
終了
戻る
次へ
キャンセル

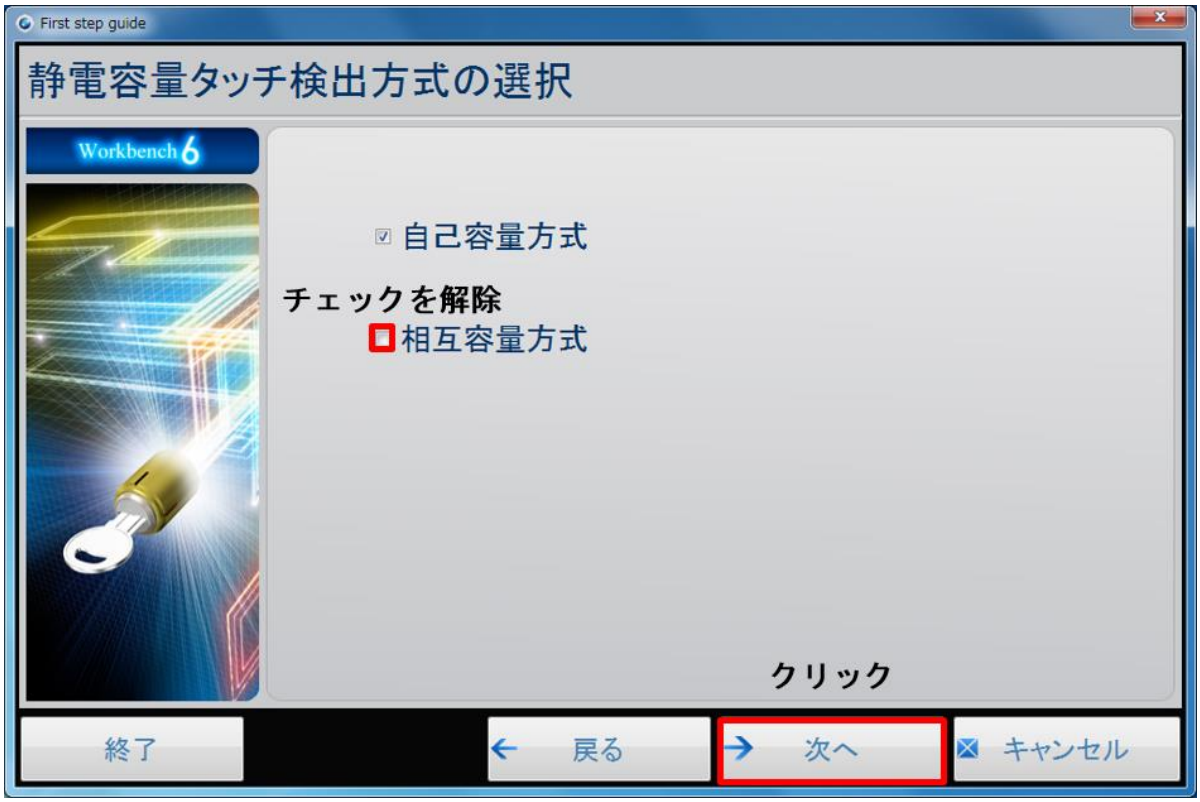

First step guide
新規プロジェクトの作成または既存プロジェクトの再開
Workbench 6
新しいプロジェクトを作成する
既存のプロジェクトを開く
プロジェクトフォルダ:
C:¥
参照...
プロジェクトフォルダは以下の形式で命名されています。
TouchAPI_YYYYMMDDhhmmss
YYYY:年
MM:月
DD:日
hh:時
mm:分
ss:秒
クリック
終了
戻る
次へ
キャンセル


First step guide
ターゲットボードとの接続方法の選択
Workbench 6
PCとターゲットボードの接続方法を選択してください。
CS+とエミュレータを使用する
e2 studioとエミュレータを使用する
e2 studioとUSBシリアル変換を使用する
サポートエミュレータは以下のとおりです。
E1、E2エミュレータLite
CS+ Utilities, CS+ for CA,CX, CS+ for CC がインストールされていません。
First step guideとWorkbenchを終了し、CS+ Utilities, CS+ for CA,CX, CS+ for CC をインストールしてください。
クリック
終了
戻る
次へ
キャンセル

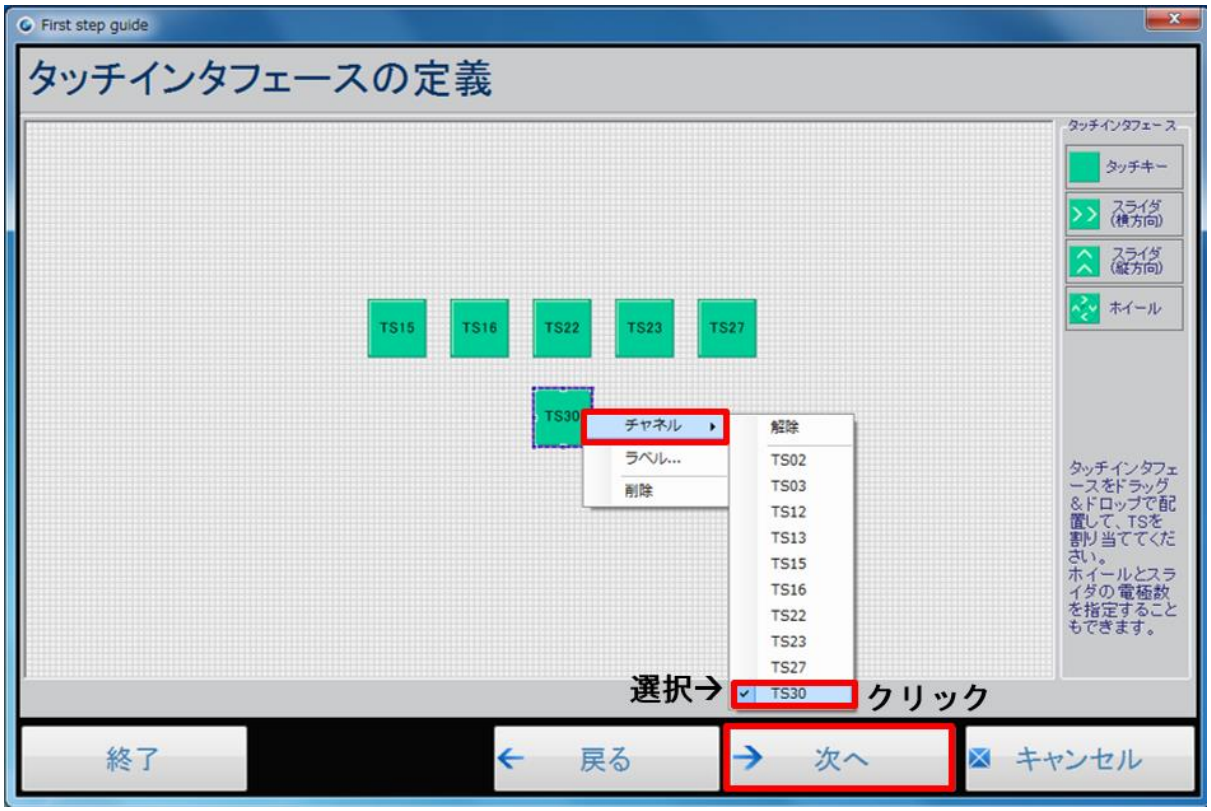

First step guide
タッチセンサマイコン選択
Workbench 6
シリーズ: RX200
グループ: RX231
品種名: R5F52315AxFM
静電容量計測端子数: 10
48 pin
64 pin
100 pin
R5F52315AxFM
R5F52315AxND
R5F52315CxFM
R5F52315CxLF
R5F52315CxND
R5F52316AxFM
R5F52316AxND
R5F52316CxFM
R5F52316CxLF
R5F52316CxND
R5F52317AxFM
R5F52317AxND
R5F52317BxFM
R5F52317BxND
R5F52318AxFM
R5F52318AxND
クリック
終了
戻る
次へ

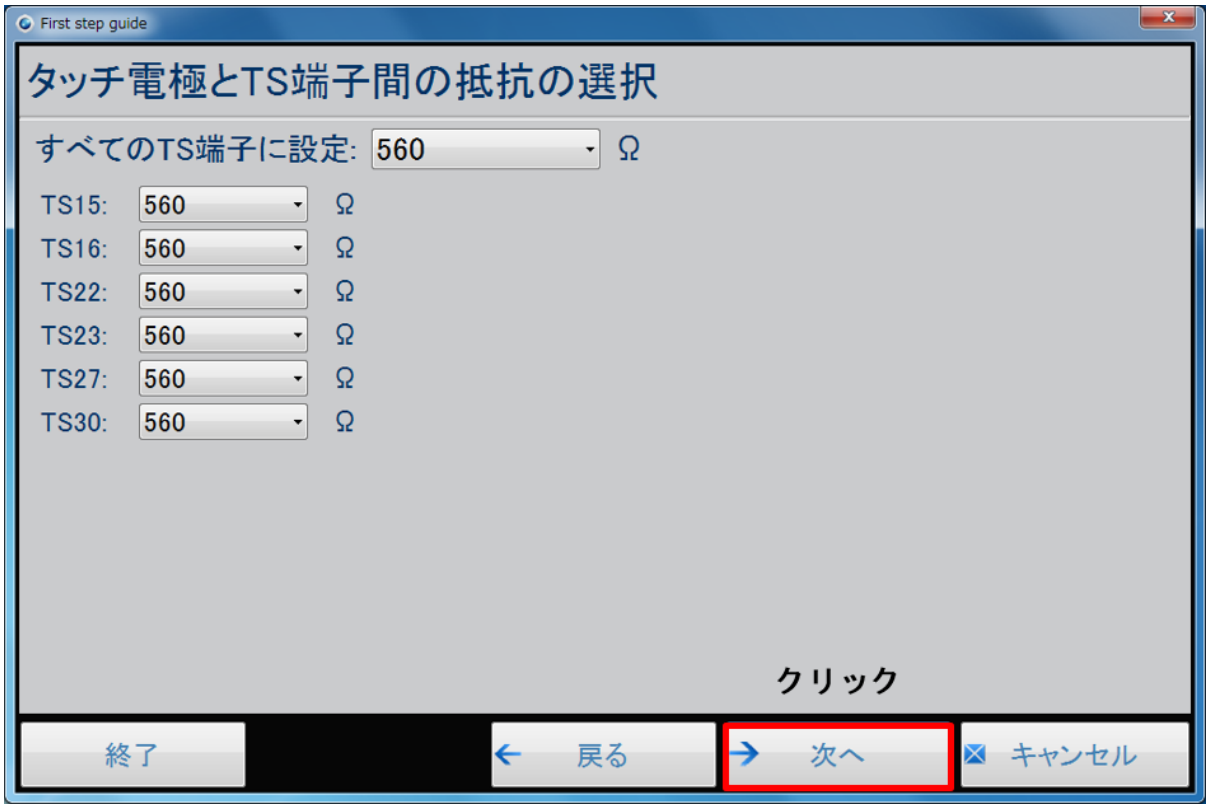

First step guide
マイコン動作クロックの選択
Workbench 6
システムクロックソース:
高速オンチップオシレータ
USB機能を使用する場合は
PLL回路を選択してください。
高速オンチップオシレータ周波数:
54 MHz
システムクロック周波数:
54 MHz
周辺モジュールクロックB周波数:
27 MHz
クリック
終了
戻る
次へ
キャンセル

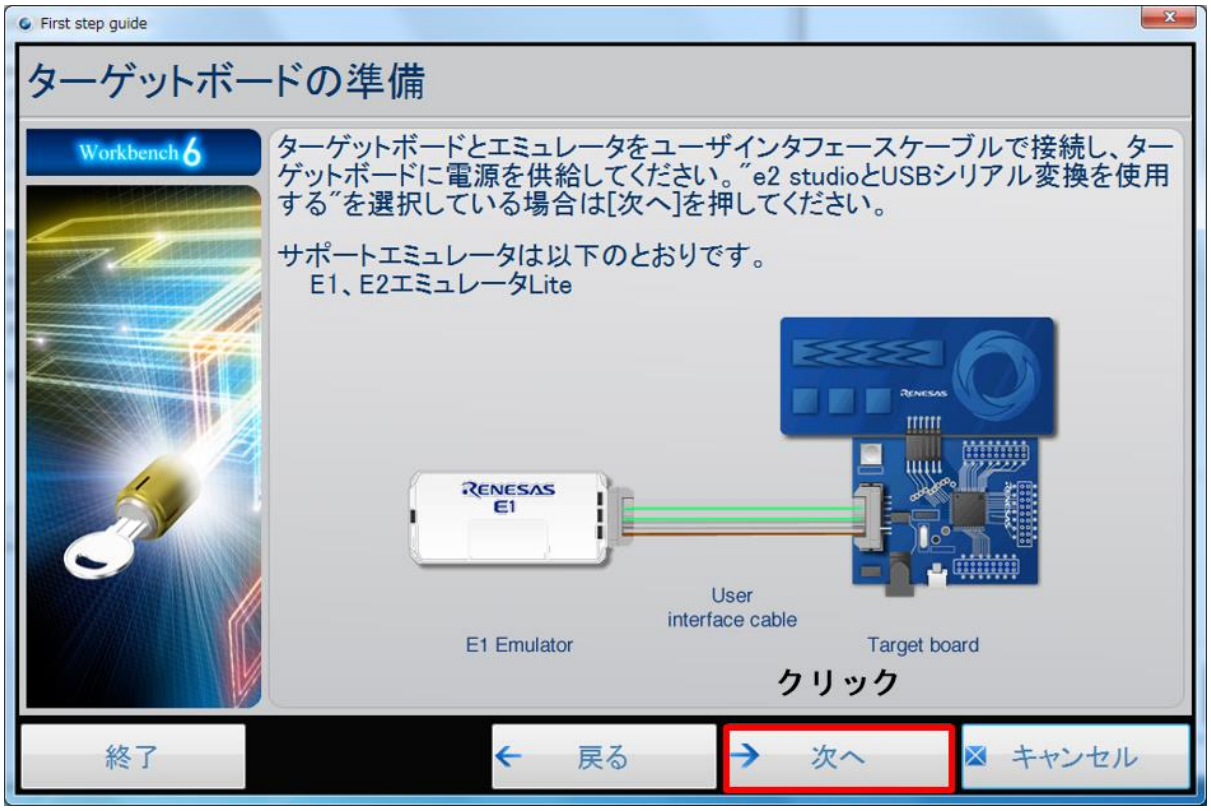

First step guide
その他の設定
Workbench 6
マイコンの動作電圧
3.3 V
USB-シリアルポート設定
シリアル通信 UART(SCI6)
クリック
終了
戻る
次へ
キャンセル

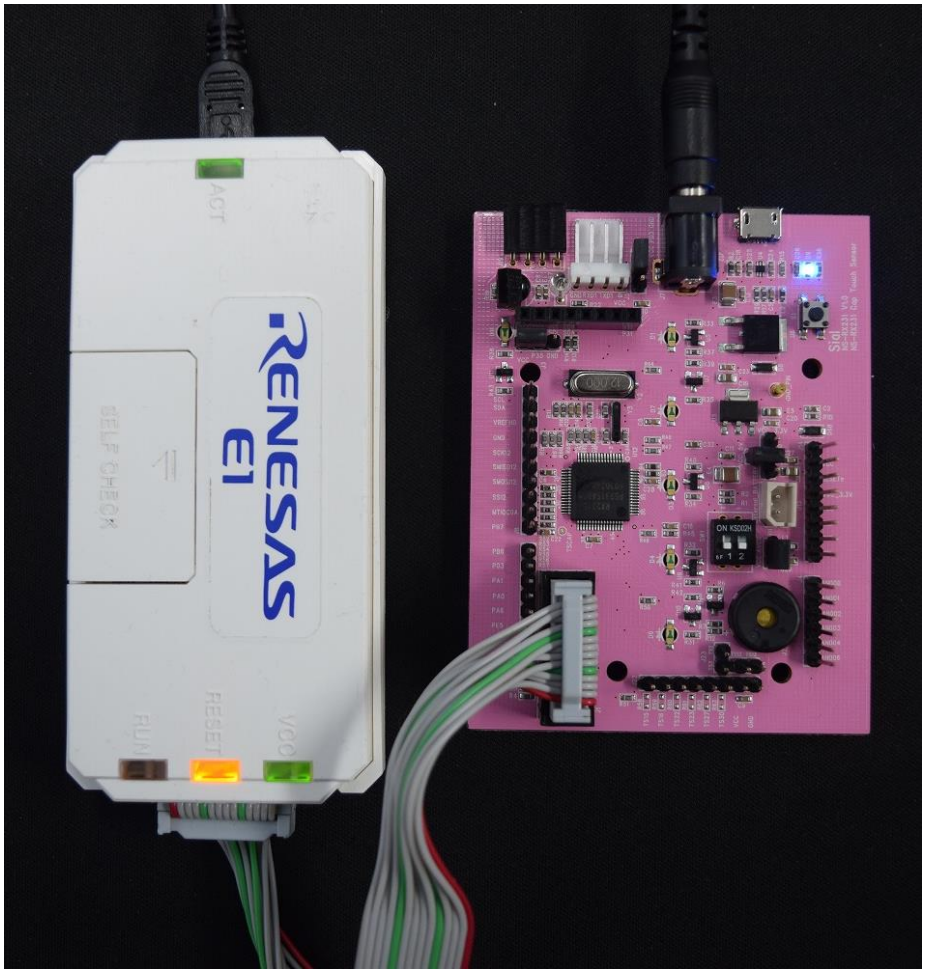

First step guide
静電容量タッチ検出方式の選択
Workbench 6
自己容量方式
チェックを解除
相互容量方式
クリック
終了
戻る
次へ
キャンセル


First step guide
プロジェクト保存フォルダの選択
Workbench 6
フォルダ:
C:¥
プロジェクト名: TouchAPI_20161213144540
参照...
プロジェクトフォルダは以下の形式で命名されます。
TouchAPI_YYYYMMDDhhmmss
YYYY:年
MM:月
DD:日
hh:時
mm:分
ss:秒
クリック
終了
戻る
次へ
キャンセル

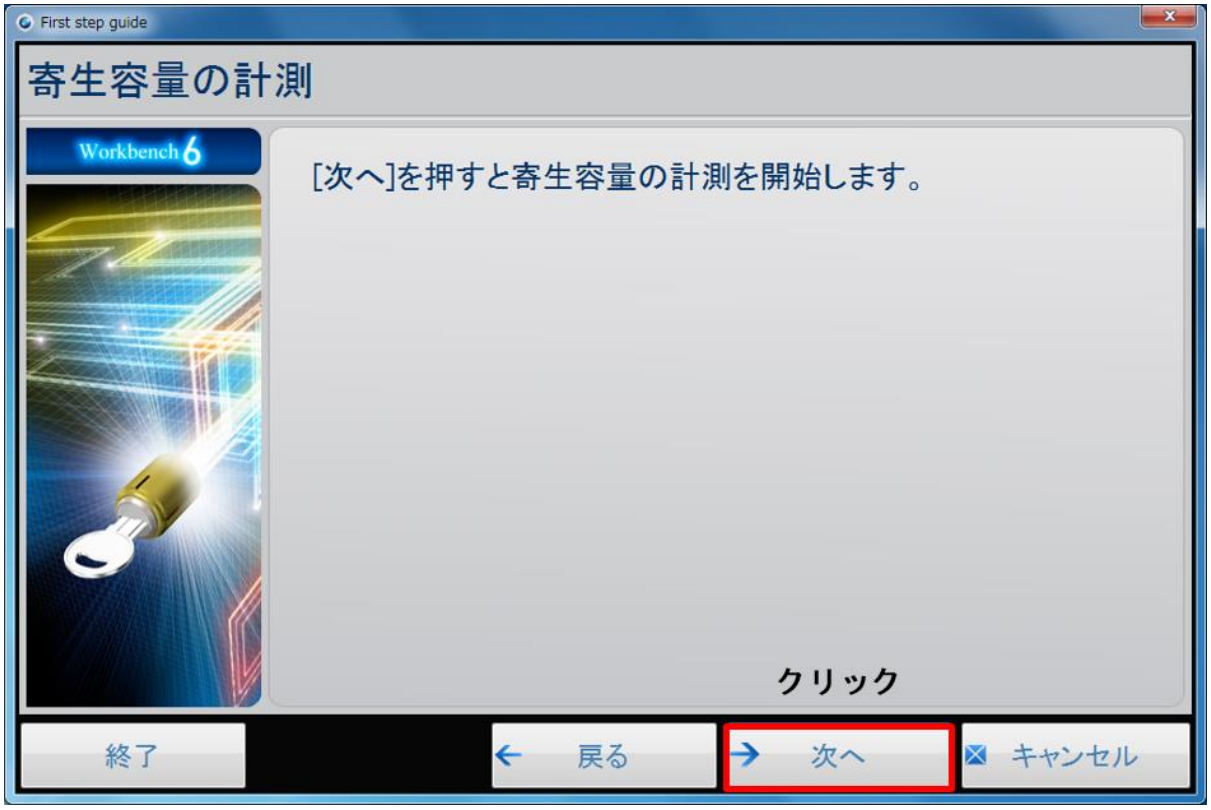

First step guide
タッチインタフェースの定義
タッチインタフェース
タッチキー
スライダ (横方向)
スライダ (縦方向)
ホイール
TS15
TS16
TS22
TS23
TS27
TS30
チャネル
ラベル...
削除
解除
TS02
TS03
TS12
TS13
TS15
TS16
TS22
TS23
TS27
TS30
タッチインタフェースをドラッグ&ドロップで配置して、TSを割り当ててください。ホイールとスライダの電極数を指定することもできます。
選択→
クリック
終了
戻る
次へ
キャンセル

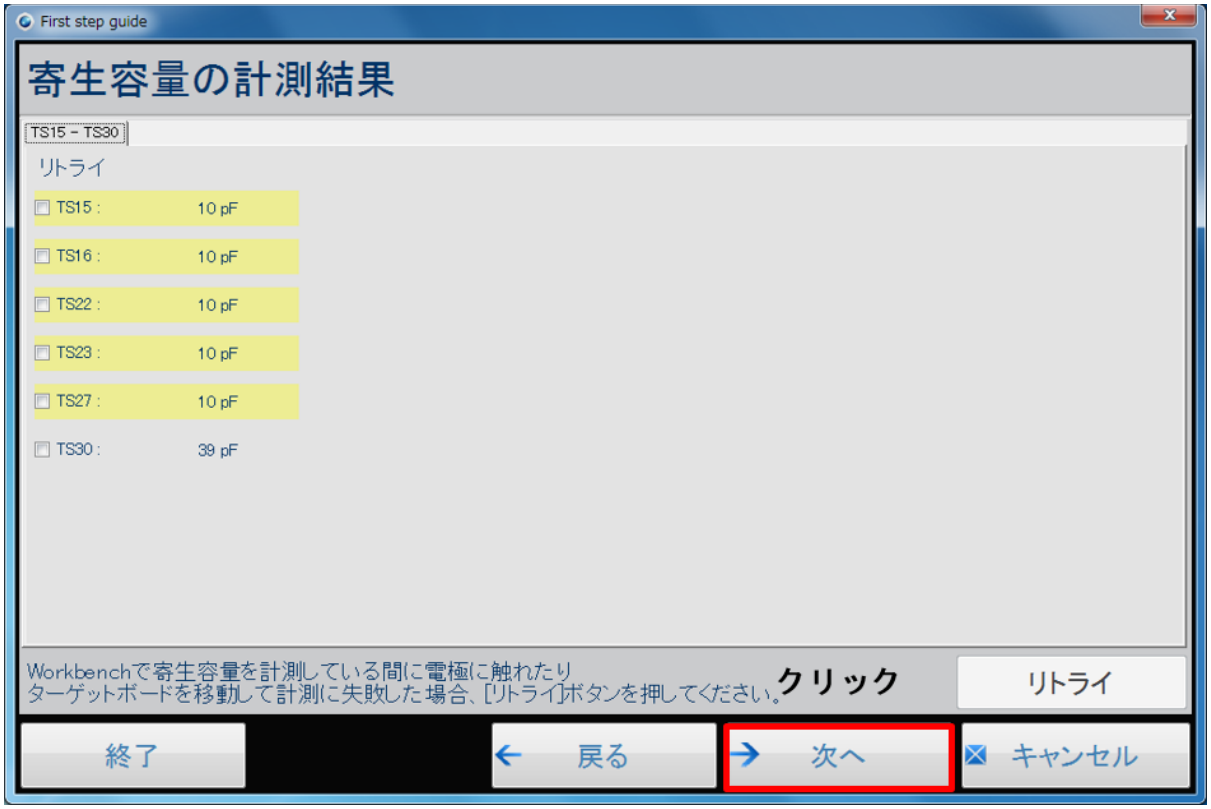

First step guide
タッチ電極とTS端子間の抵抗の選択
すべてのTS端子に設定: 560 Ω
TS15: 560 Ω
TS16: 560 Ω
TS22: 560 Ω
TS23: 560 Ω
TS27: 560 Ω
TS30: 560 Ω
クリック
終了
戻る
次へ
キャンセル


First step guide
ターゲットボードの準備
Workbench 6
ターゲットボードとエミュレータをユーザインタフェースケーブルで接続し、ターゲットボードに電源を供給してください。"e2 studioとUSBシリアル変換を使用する"を選択している場合は[次へ]を押してください。
サポートエミュレータは以下のとおりです。
E1、E2エミュレータLite
RENESAS
E1
User
interface cable
E1 Emulator
Target board
クリック
終了
戻る
次へ
キャンセル

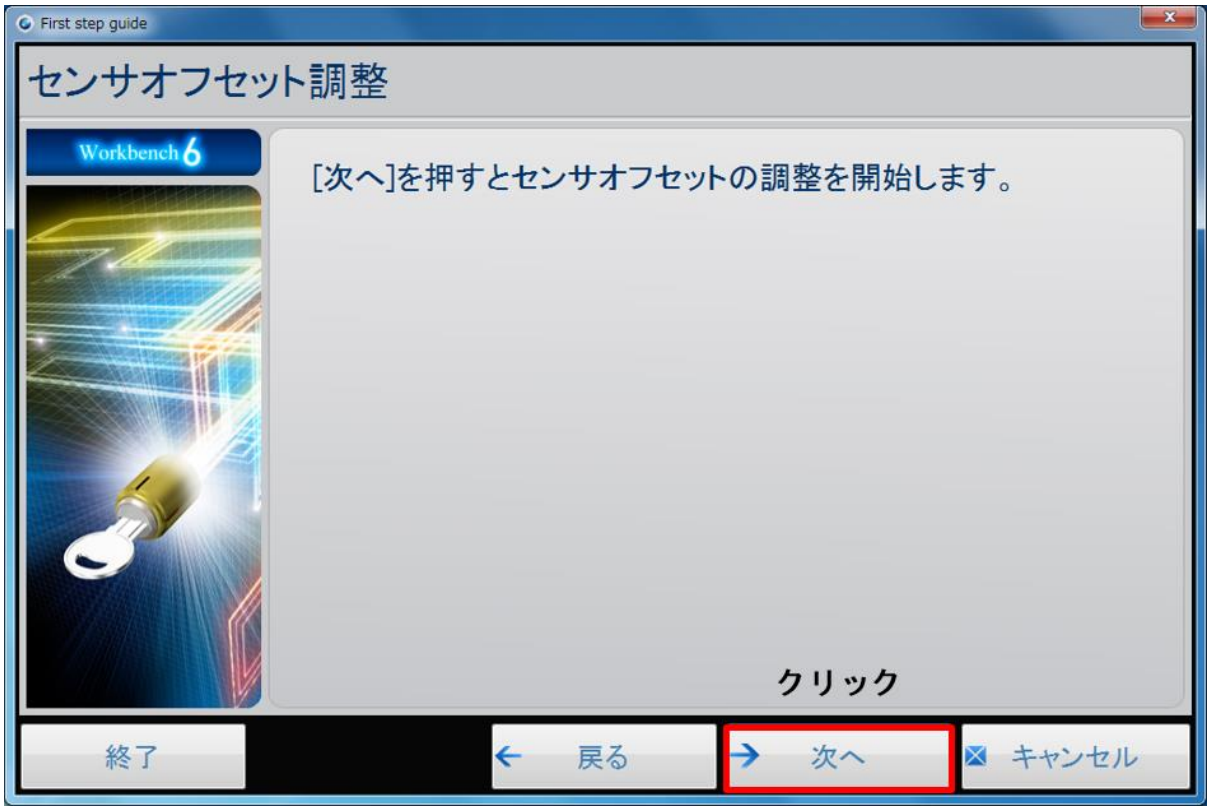

RENESAS
E1

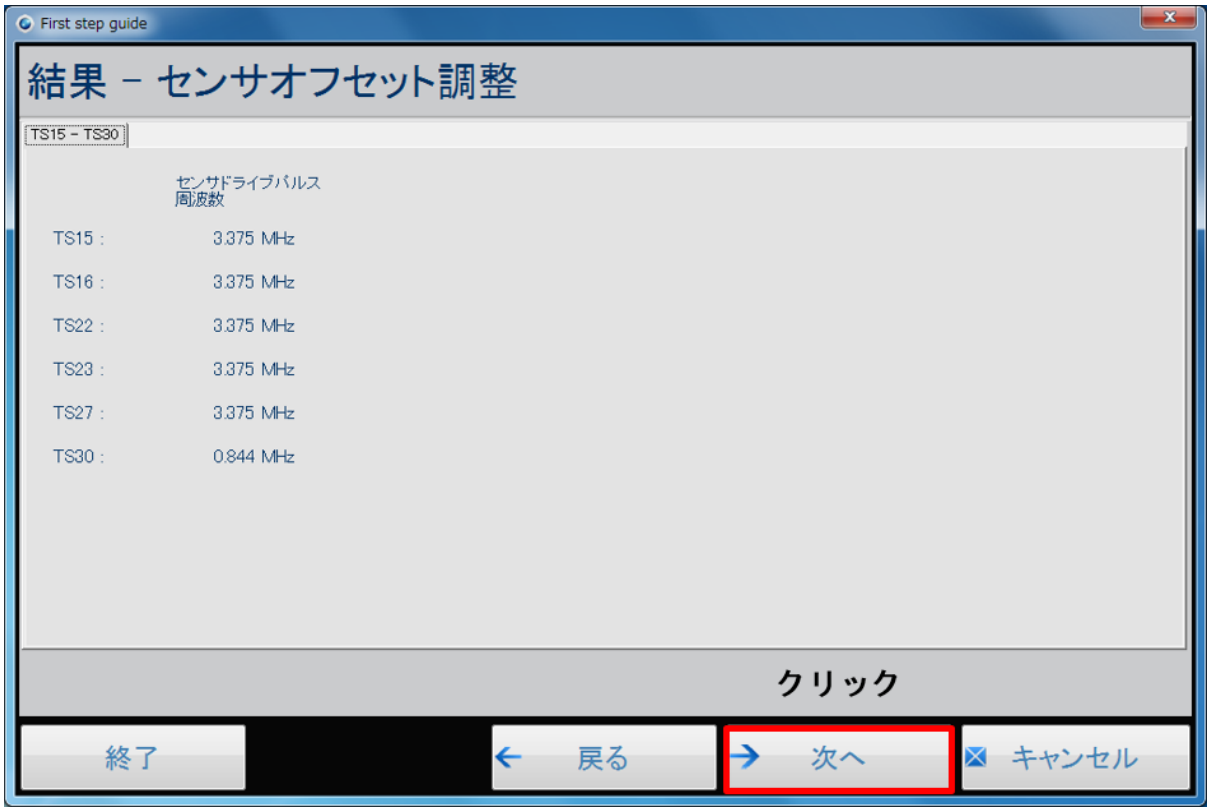

First step guide
ターゲットボードの起動
Workbench 6
ターゲットプロジェクトオープン
ビルド完了
ファームウェアのダウンロード完了
ファームウェアの実行完了
続ける場合は[次へ]、
ウィザードを中断する場合は[キャンセル]を押してください。
クリック
終了
戻る
次へ
キャンセル


First step guide
寄生容量の計測
Workbench 6
[次へ]を押すと寄生容量の計測を開始します。
クリック
終了
戻る
次へ
キャンセル

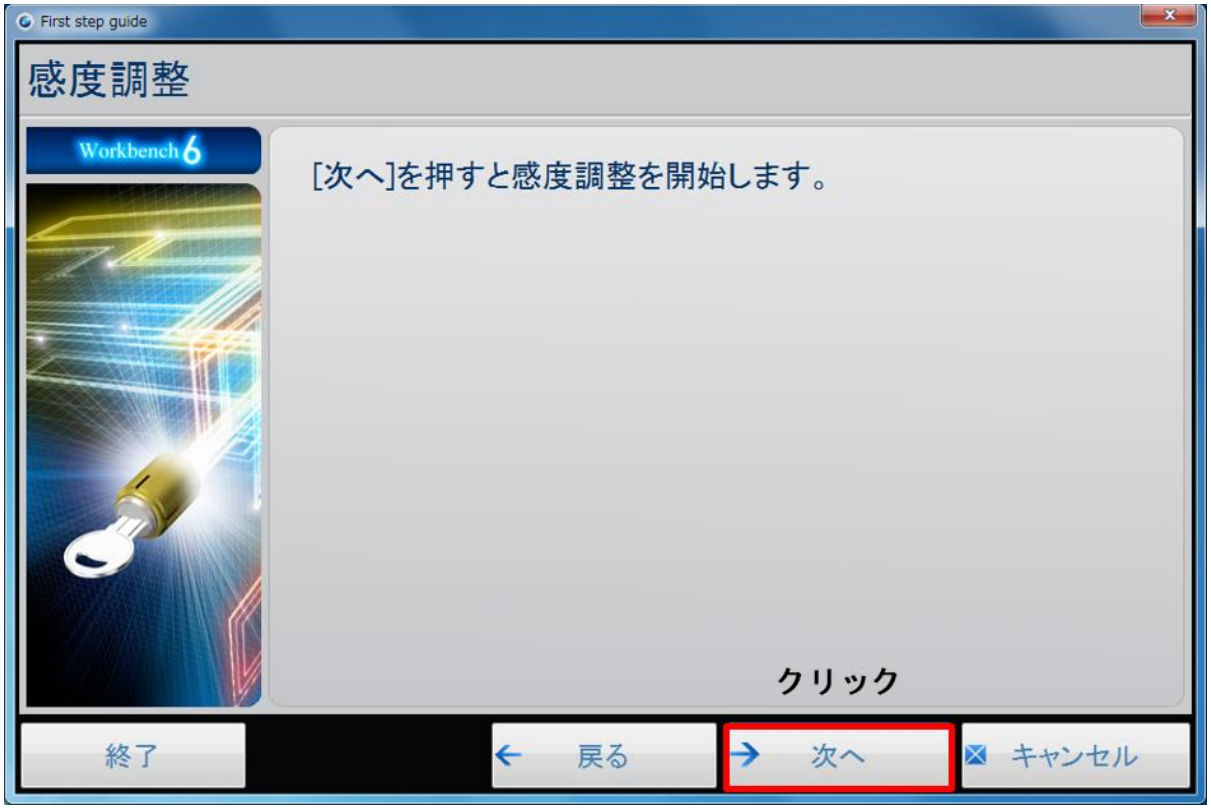


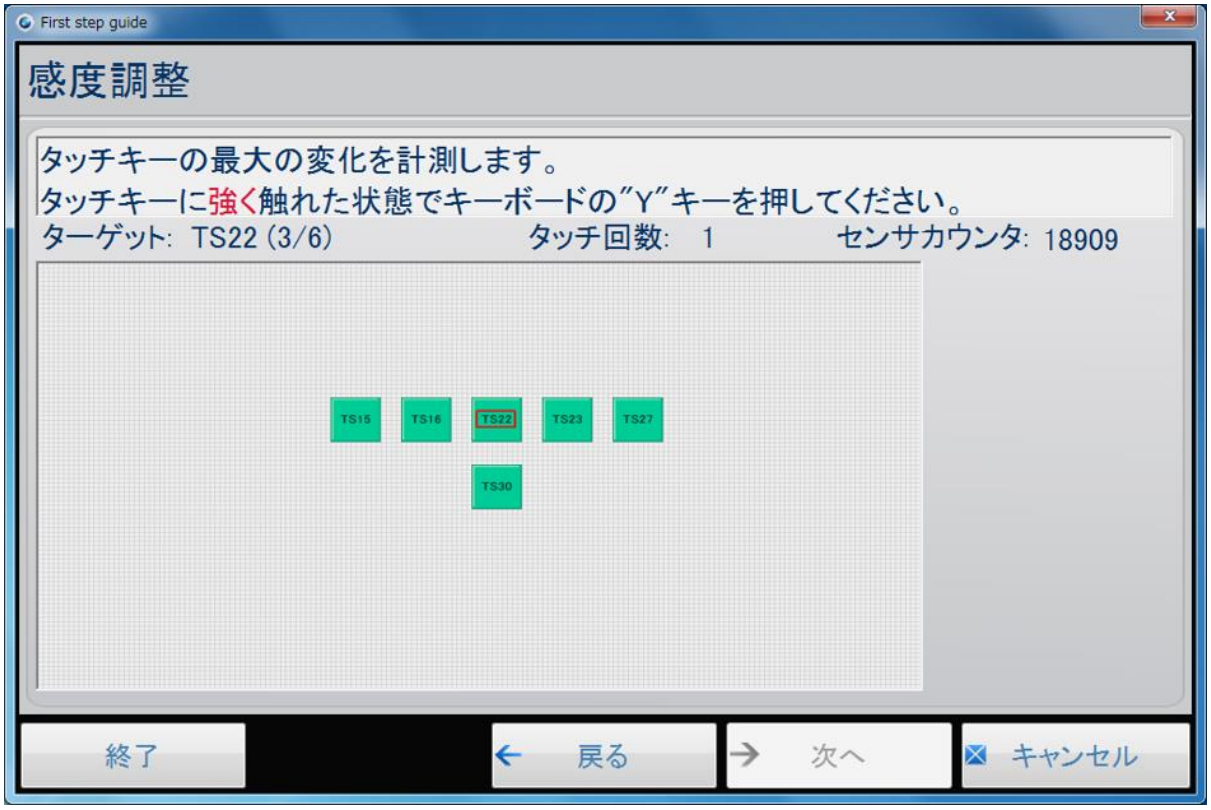
容量値は一例で，この値とは異なることがあります

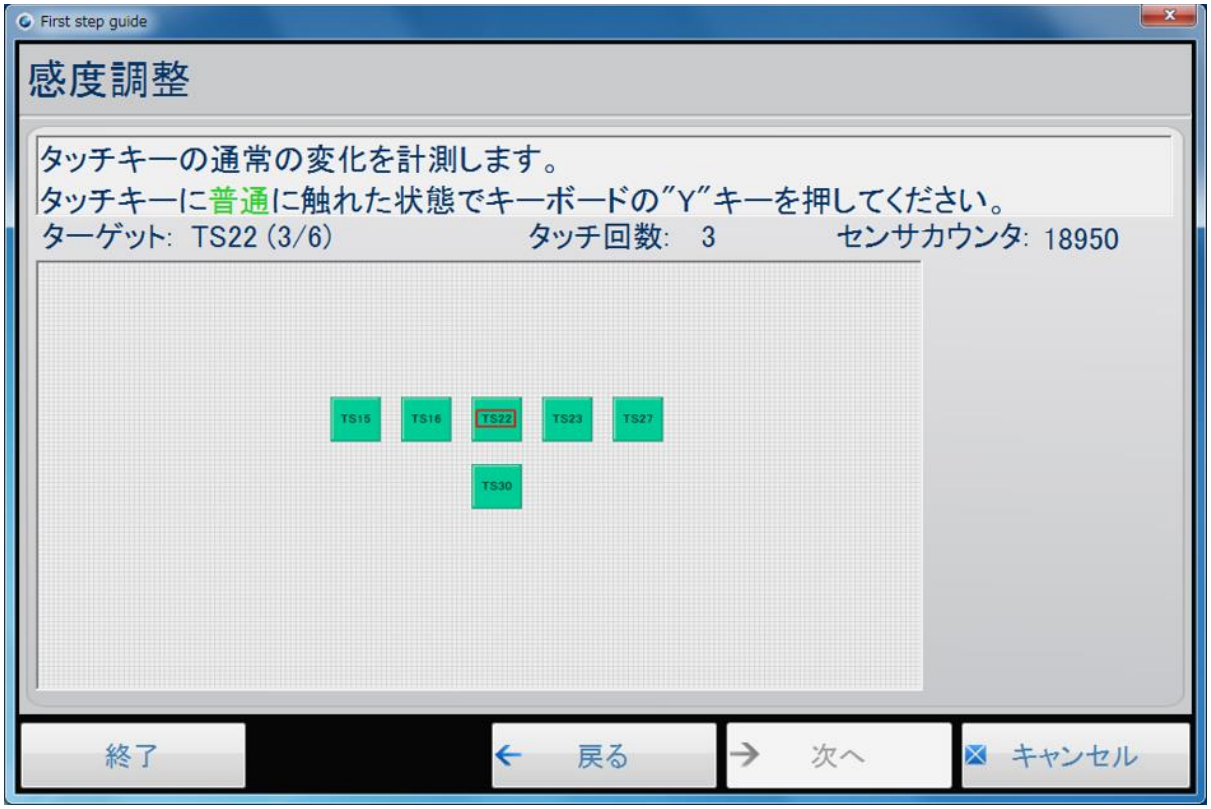


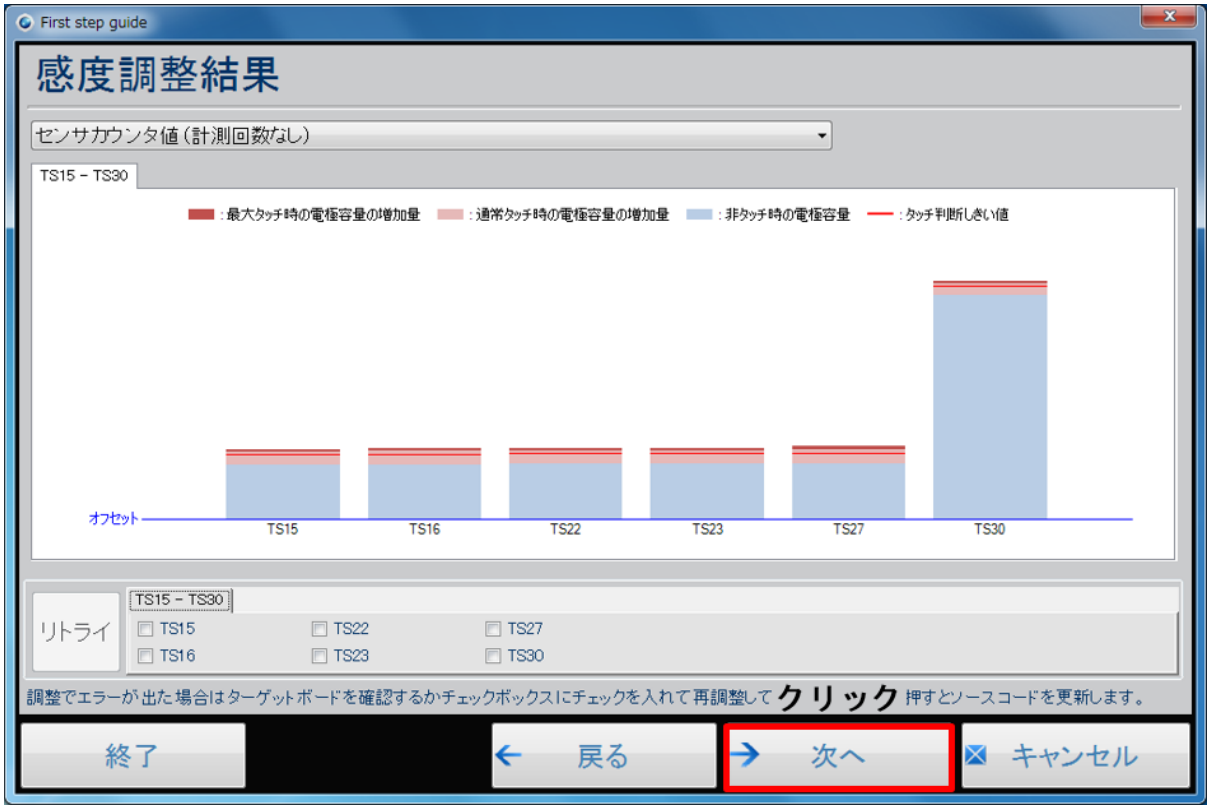


周波数は一例で，この値とは異なることがあります


First step guide
感度調整
Workbench 6
[次へ]を押すと感度調整を開始します。
クリック
終了
戻る
次へ
キャンセル


First step guide
感度調整
タッチキーの最大の変化を計測します。
タッチキーに強く触れた状態でキーボードの"Y"キーを押してください。
ターゲット: TS22 (3/6)
タッチ回数: 1
センサカウンタ: 18909
TS15
TS16
TS22
TS23
TS27
TS30
終了
戻る
次へ
キャンセル


First step guide
感度調整
タッチキーの通常の変化を計測します。
タッチキーに普通に触れた状態でキーボードの"Y"キーを押してください。
ターゲット: TS22 (3/6)
タッチ回数: 3
センサカウンタ: 18950
TS15
TS16
TS22
TS23
TS27
TS30
終了
戻る
次へ
キャンセル


First step guide
感度調整結果
センサカウンタ値(計測回数なし)
TS15 - TS30
: 最大タッチ時の電極容量の増加量
: 通常タッチ時の電極容量の増加量
: 非タッチ時の電極容量
: タッチ判断しきい値
オフセット
TS15
TS16
TS22
TS23
TS27
TS30
リトライ
TS15 - TS30
TS15
TS16
TS22
TS23
TS27
TS30
調整でエラーが出た場合はターゲットボードを確認するかチェックボックスにチェックを入れて再調整して クリック 押すとソースコードを更新します。
終了
戻る
次へ
キャンセル


First step guide
ターゲットボードの起動
Workbench 6
ターゲットプロジェクトオープン
ビルド完了
ファームウェアのダウンロード完了
ファームウェアの実行完了
[完了]を押してこのウィザードを閉じてください。
クリック
終了
戻る
完了
キャンセル

2 ソース編集

2.1 レジスタ

最初にLED制御のためのレジスタから見てみます。 詳細はRX231グループ ユーザーズマニュアル ハードウェア編を参照してください。

21.3.1 ポート方向レジスタ（PDR）

アドレス PORT0.PDR 0008 C000h, PORT1.PDR 0008 C001h, PORT2.PDR 0008 C002h, PORT3.PDR 0008 C003h, PORT4.PDR 0008 C004h, PORT5.PDR 0008 C005h, PORTA.PDR 0008 C00Ah, PORTB.PDR 0008 C00Bh, PORTC.PDR 0008 C00Ch, PORTD.PDR 0008 C00Dh, PORTE.PDR 0008 C00Eh, PORTH.PDR 0008 C011h, PORTJ.PDR 0008 C012h

	b7	b6	b5	b4	b3	b2	b1	b0
	B7	B6	B5	B4	B3	B2	B1	B0
リセット後の値	0	0	0	0	0	0	0	0

ビット	シンボル	ビット名	機能	R/W
b0	B0	Pm0方向制御ビット	0：入力（入力ポートとして機能） 1：出力（出力ポートとして機能）	R/W
b1	B1	Pm1方向制御ビット		R/W
b2	B2	Pm2方向制御ビット		R/W
b3	B3	Pm3方向制御ビット		R/W
b4	B4	Pm4方向制御ビット		R/W
b5	B5	Pm5方向制御ビット		R/W
b6	B6	Pm6方向制御ビット		R/W
b7	B7	Pm7方向制御ビット		R/W

m = 0 ～ 5, A ～ E, H, J

ポート方向レジスタ(PDR)は汎用入出力ポートを入力で使用するか出力で使用するかを指定します。初期値は0であり、0に設定すれば入力、1に設定すれば出力になります。

21.3.2 ポート出力データレジスタ（PODR）

アドレス PORT0.PODR 0008 C020h, PORT1.PODR 0008 C021h, PORT2.PODR 0008 C022h, PORT3.PODR 0008 C023h, PORT4.PODR 0008 C024h, PORT5.PODR 0008 C025h, PORTA.PODR 0008 C02Ah, PORTB.PODR 0008 C02Bh, PORTC.PODR 0008 C02Ch, PORTD.PODR 0008 C02Dh, PORTE.PODR 0008 C02Eh, PORTH.ODR 0008 C031h, PORTJ.PODR 0008 C032h

	b7	b6	b5	b4	b3	b2	b1	b0
	B7	B6	B5	B4	B3	B2	B1	B0
リセット後の値	0	0	0	0	0	0	0	0

ビット	シンボル	ビット名	機能	R/W
b0	B0	Pm0出力データ格納ビット	出力データ格納	R/W
b1	B1	Pm1出力データ格納ビット		R/W
b2	B2	Pm2出力データ格納ビット		R/W
b3	B3	Pm3出力データ格納ビット		R/W
b4	B4	Pm4出力データ格納ビット		R/W
b5	B5	Pm5出力データ格納ビット		R/W
b6	B6	Pm6出力データ格納ビット		R/W
b7	B7	Pm7出力データ格納ビット		R/W

m = 0 ～ 5, A ～ E, H, J

ポート出力データ レジスタ(PODR)は汎用出力ポートとして使用する端子の出力データを設定するレジスタです。 PMR,PDRレジスタの設定が正しければ，PODRが1である間は端子からHIGHを出力します。

21.3.3 ポート入力データレジスタ（PIDR）

アドレス PORT0.PIDR 0008 C040h, PORT1.PIDR 0008 C041h, PORT2.PIDR 0008 C042h, PORT3.PIDR 0008 C043h, PORT4.PIDR 0008 C044h, PORT5.PIDR 0008 C045h, PORTA.PIDR 0008 C04Ah, PORTB.PIDR 0008 C04Bh, PORTC.PIDR 0008 C04Ch, PORTD.PIDR 0008 C04Dh, PORTE.PIDR 0008 C04Eh, PORTH.PIDR 0008 C051h, PORTJ.PIDR 0008 C052h

	b7	b6	b5	b4	b3	b2	b1	b0
	B7	B6	B5	B4	B3	B2	B1	B0
リセット後の値	x	x	x	x	x	x	x	x

x：不定

ビット	シンボル	ビット名	機能	R/W
b0	B0	Pm0ビット	ポートの端子状態を反映	R
b1	B1	Pm1ビット		R
b2	B2	Pm2ビット		R
b3	B3	Pm3ビット		R
b4	B4	Pm4ビット		R
b5	B5	Pm5ビット		R
b6	B6	Pm6ビット		R
b7	B7	Pm7ビット		R

m = 0 ～ 5, A ～ E, H, J

ポート入力データ レジスタ(PIDR)はポート端子の状態が読み取るレジスタです。 読み取りだけ可能で、PDR,PMRレジスタの設定とは関係なく端子の現在のデータを読むことができます。

21.3.4 ポートモードレジスタ（PMR）

アドレス PORT0.PMR 0008 C060h, PORT1.PMR 0008 C061h, PORT2.PMR 0008 C062h, PORT3.PMR 0008 C063h, PORT4.PMR 0008 C064h, PORT5.PMR 0008 C065h, PORTA.PMR 0008 C06Ah, PORTB.PMR 0008 C06Bh, PORTC.PMR 0008 C06Ch, PORTD.PMR 0008 C06Dh, PORTE.PMR 0008 C06Eh, PORTH.PMR 0008 C071h, PORTJ.PMR 0008 C072h

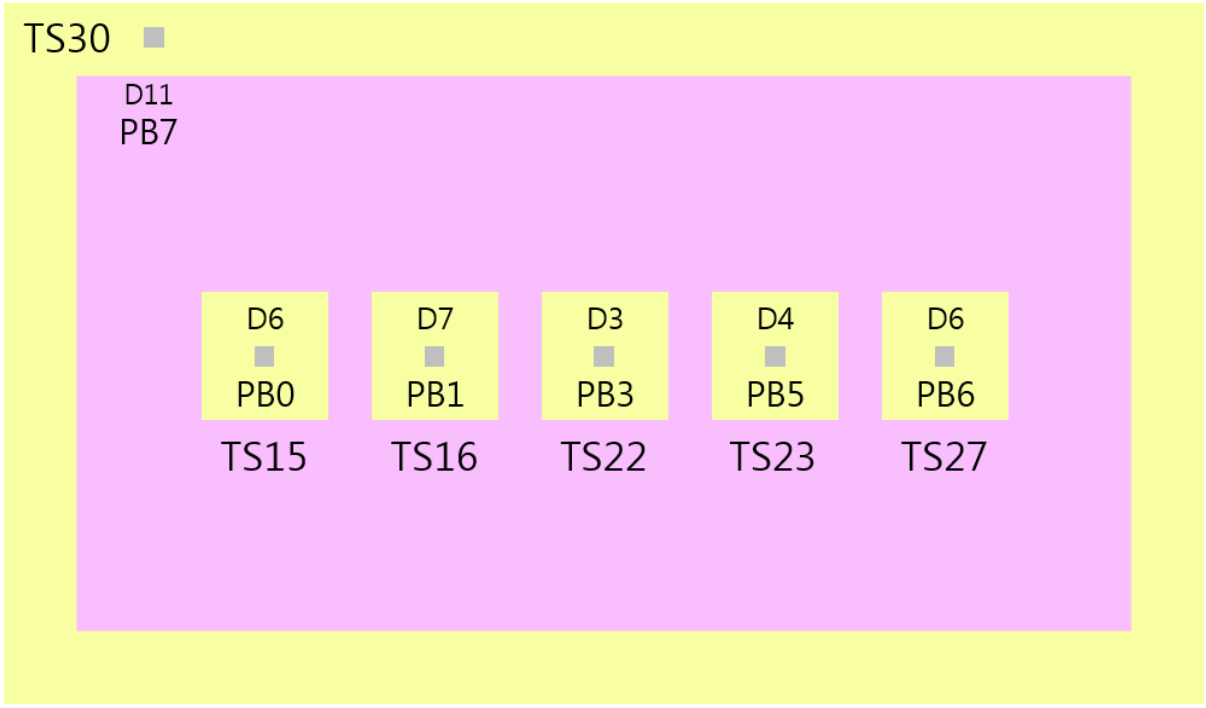
	b7	b6	b5	b4	b3	b2	b1	b0
	B7	B6	B5	B4	B3	B2	B1	B0
リセット後の値	0	0	0	0	0	0	0	0

ビット	シンボル	ビット名	機能	R/W
b0	B0	Pm0端子モード制御ビット	0：汎用入出力ポートとして使用 1：周辺機能として使用	R/W
b1	B1	Pm1端子モード制御ビット		R/W
b2	B2	Pm2端子モード制御ビット		R/W
b3	B3	Pm3端子モード制御ビット		R/W
b4	B4	Pm4端子モード制御ビット		R/W
b5	B5	Pm5端子モード制御ビット		R/W
b6	B6	Pm6端子モード制御ビット		R/W
b7	B7	Pm7端子モード制御ビット		R/W

m = 0 ～ 5, A ～ E, H, J

ポート モード レジスタ(PMR)の初期値は0です。1に設定すれば割り込みや通信などの周辺機能で端子を使用し、0に設定すればPODRレジスタを通じて制御ができるようになります。

2.2 ソースコード




```
if (_0_SUCCESS == R_Set_Cap_Touch_Result_Create( method ))
{
    ts_result = R_Get_Cap_Touch_Result( method );

    if (1 == (ts_result.button[0] >> 15))       //TS15
    {
        PORTB.PODR.BYTE = 0x01; //PB0 0000 0001
    }
    else if(1 == (ts_result.button[1] >> 0))    //TS16
    {
        PORTB.PODR.BYTE = 0x02; //PB1 0000 0010
    }
    else if(1 == (ts_result.button[1] >> 6))    //TS22
    {
        PORTB.PODR.BYTE = 0x08; //PB3 0000 1000
    }
    else if(1 == (ts_result.button[1] >> 7))    //TS23
    {
        PORTB.PODR.BYTE = 0x20; //PB5 0010 0000
    }
    else if(1 == (ts_result.button[1] >> 11))   //TS27
    {
        PORTB.PODR.BYTE = 0x40; //PB6 0100 0000
    }
    else if(1 == (ts_result.button[1] >> 14))   //TS30
    {
        PORTB.PODR.BYTE = 0x80; //PB7 1000 0000
    }
}
```

ts_resultはbutton,slider,wheelからなります。 そのうちのbuttonについて説明します。

ts_result.buttonは[0～2]値を有していて、その値は下式で表されます。

$$\text{ts\_result.button}[n'] = 2^{TSn-16\times n'}$$

ts_result.button[0]	TS0 ~ TS15
ts_result.button[1]	TS16 ~ TS31
ts_result.button[2]	TS32 ~ TS47

TS15がタッチされたと認識されたとき、button[0]の値は2^{15}（=32768）になります。

```
if(1 == (ts_result.button[0] >> 15))        //TS15
{
    PORTB.PODR.BYT
}
else if(1 == (ts_r
{
    PORTB.PODR.BYT
}
```

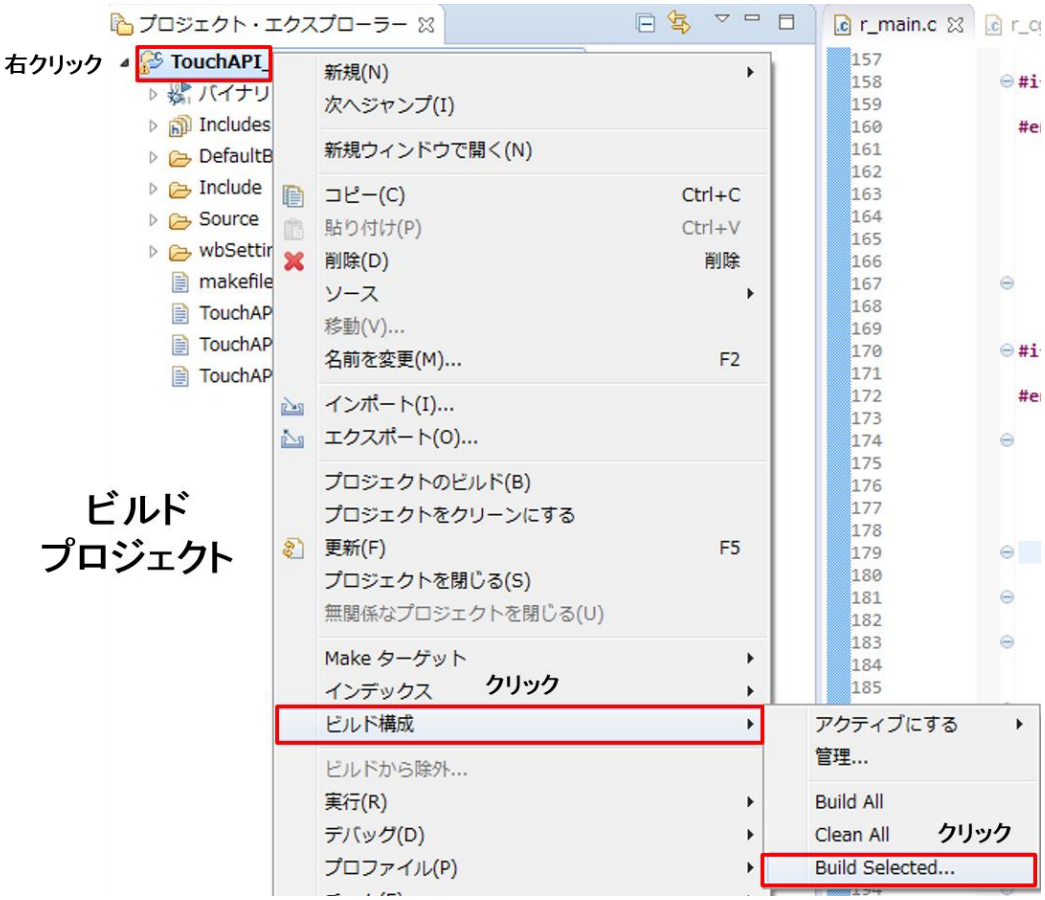
式	タイプ	値
ts_result.button	uint16_t [3]	0x5d0 <ts_result>
ts_result.button[0]	uint16_t	32768
ts_result.button[1]	uint16_t	0
ts_result.button[2]	uint16_t	0

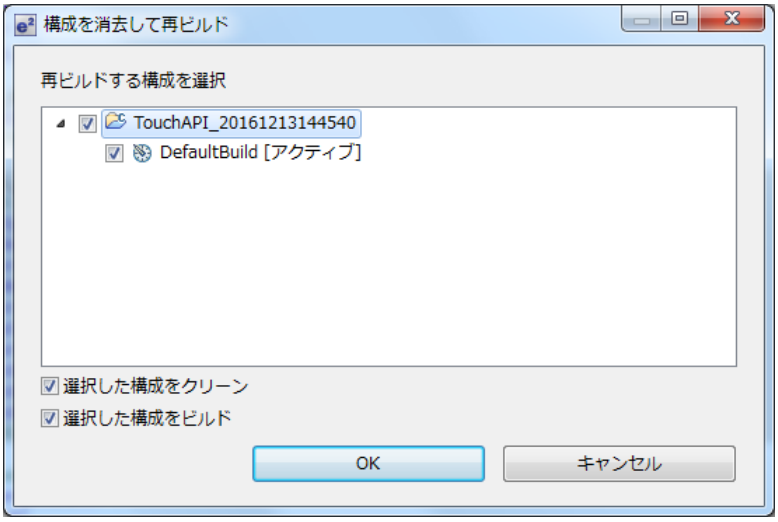
TS27がタッチされたと認識されたとき、button[1]の値は2^{11}（=2048）になります。

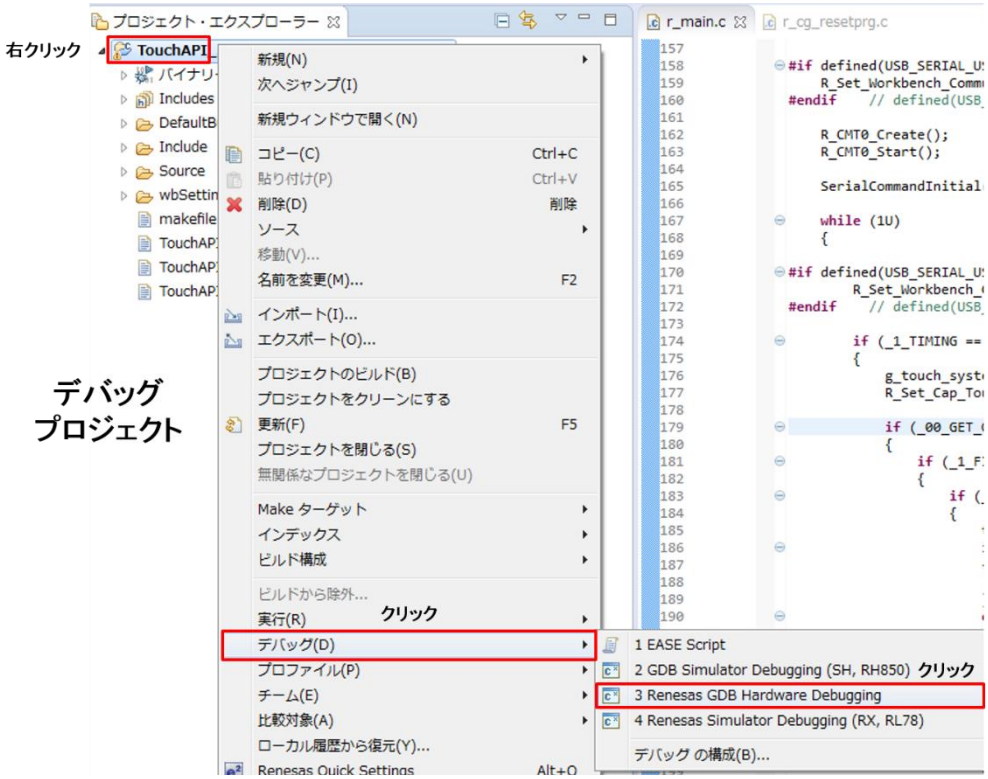
```
else if(1 == (ts_result.button[1] >> 11))   //TS27
{
    PORTB.PODR.BYTE = 0x
}
else if(1 == (ts_result.
{
    PORTB.PODR.BYTE = 0x
}
```

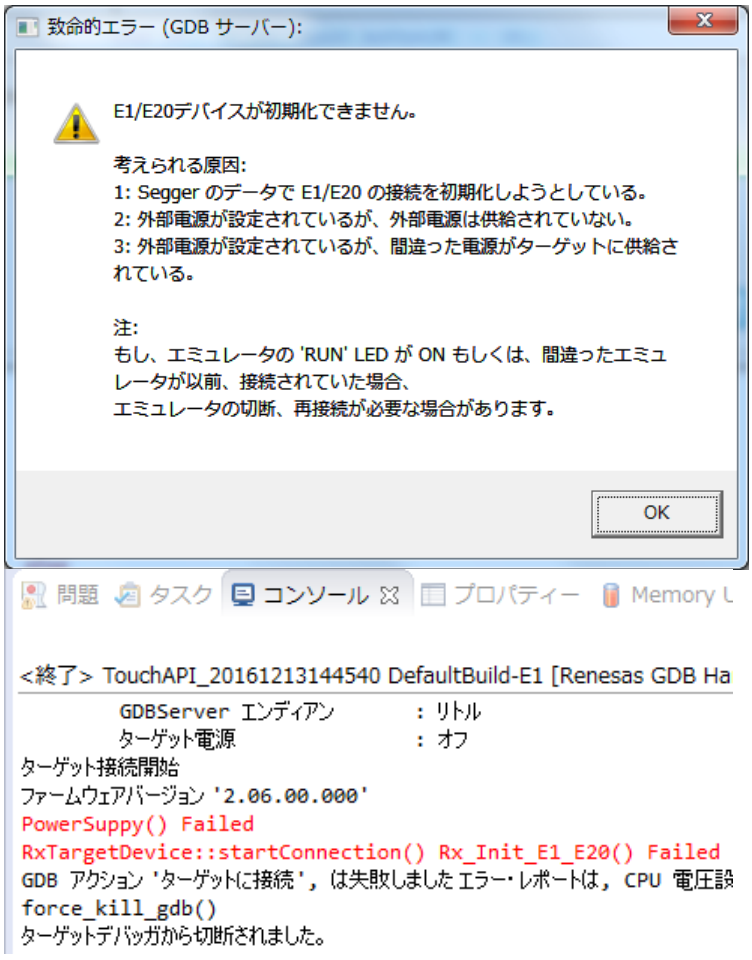
式	タイプ	値
ts_result.button	uint16_t [3]	0x5d0 <ts_result>
ts_result.button[0]	uint16_t	0
ts_result.button[1]	uint16_t	2048
ts_result.button[2]	uint16_t	0

3 プロジェクトの書き込み

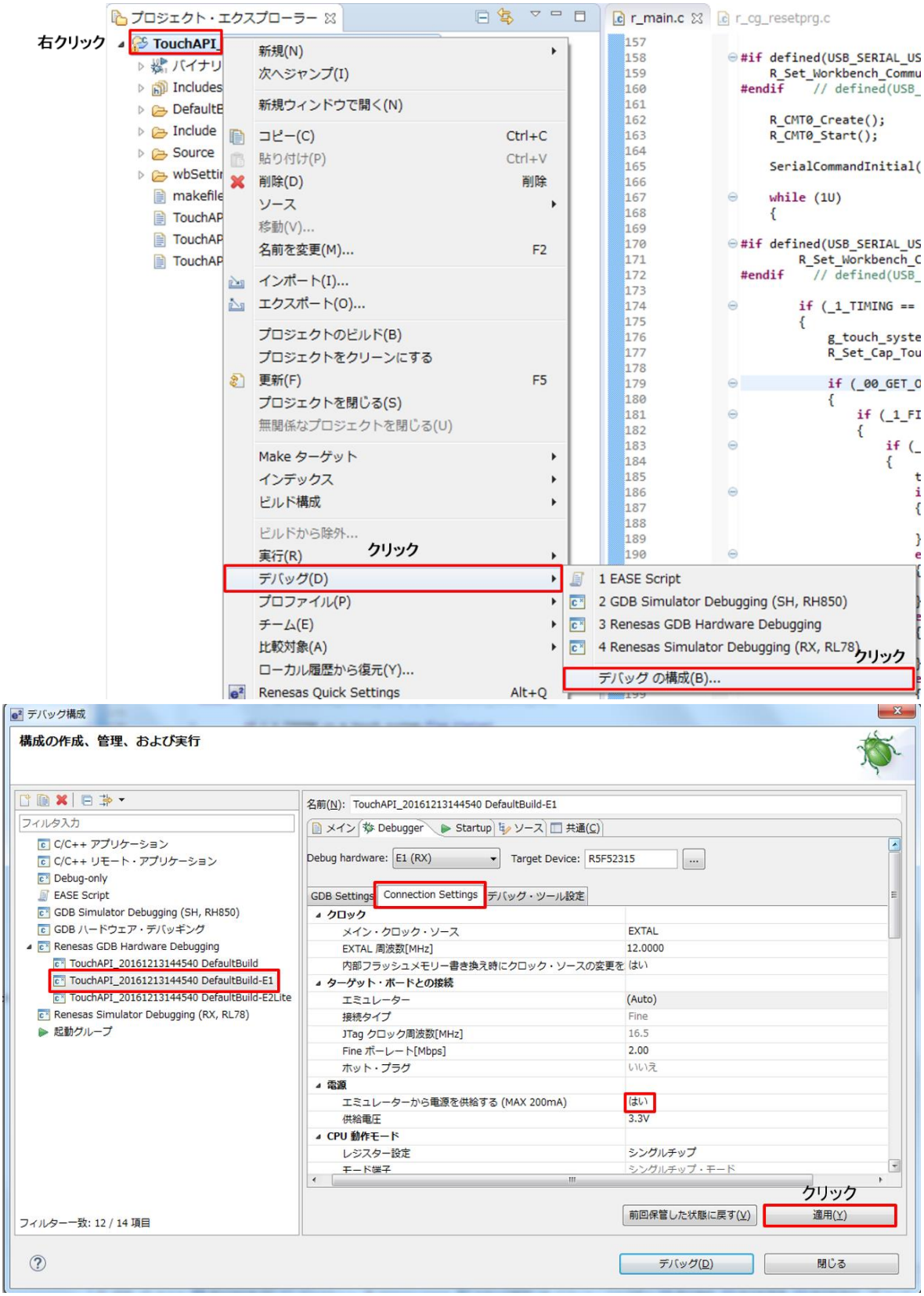






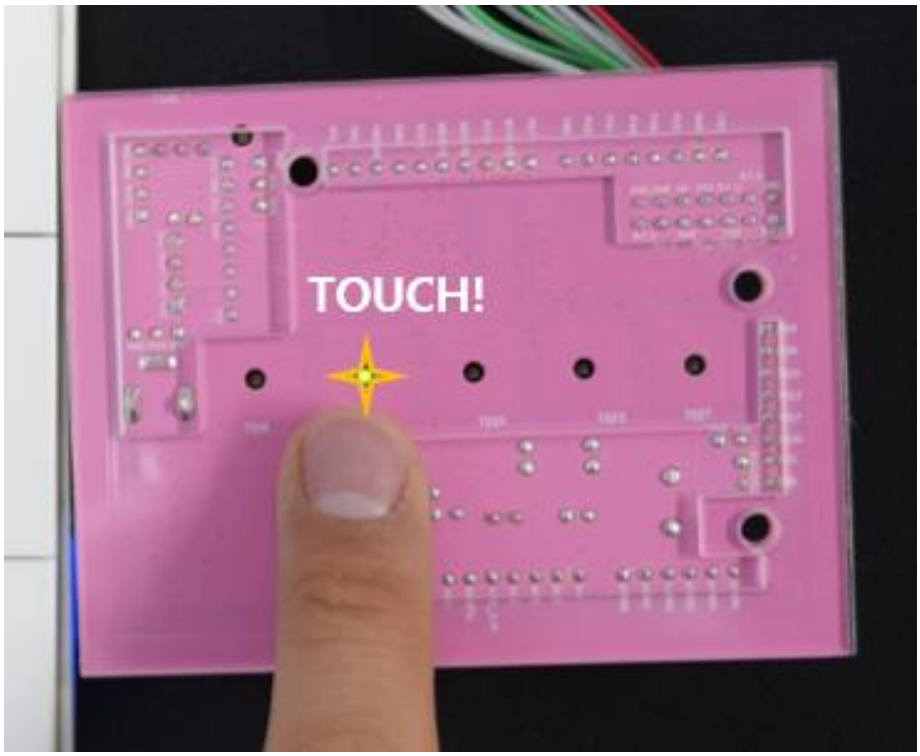


デバッグ中に、上記のようなエラーメッセージがでたときは、次の設定を参考にしてください。



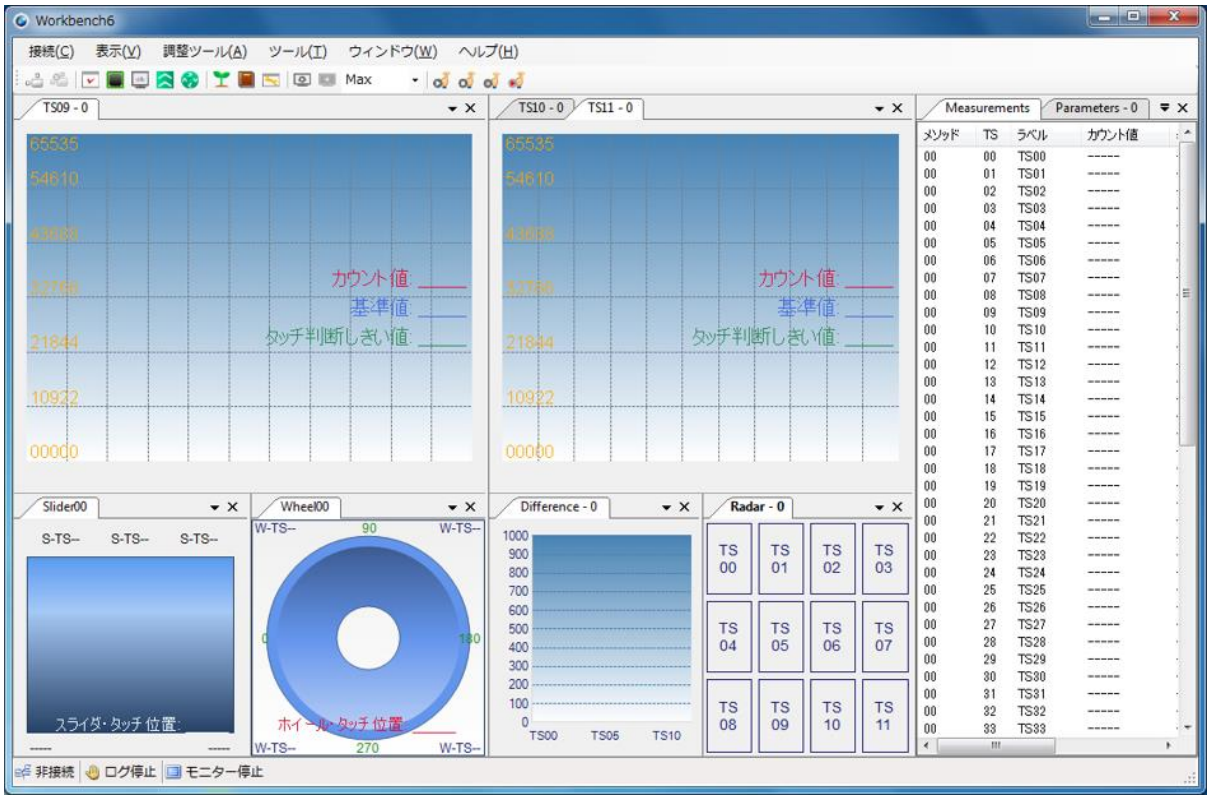
プロジェクトを作った直後には上図に示すようにエミュレータを通じて電源を供給するオプションがいいえになっています。 はいに変更すれば追加電源(アダプタ、マイクロUSBなど)がなくてもデバッグを進められます。

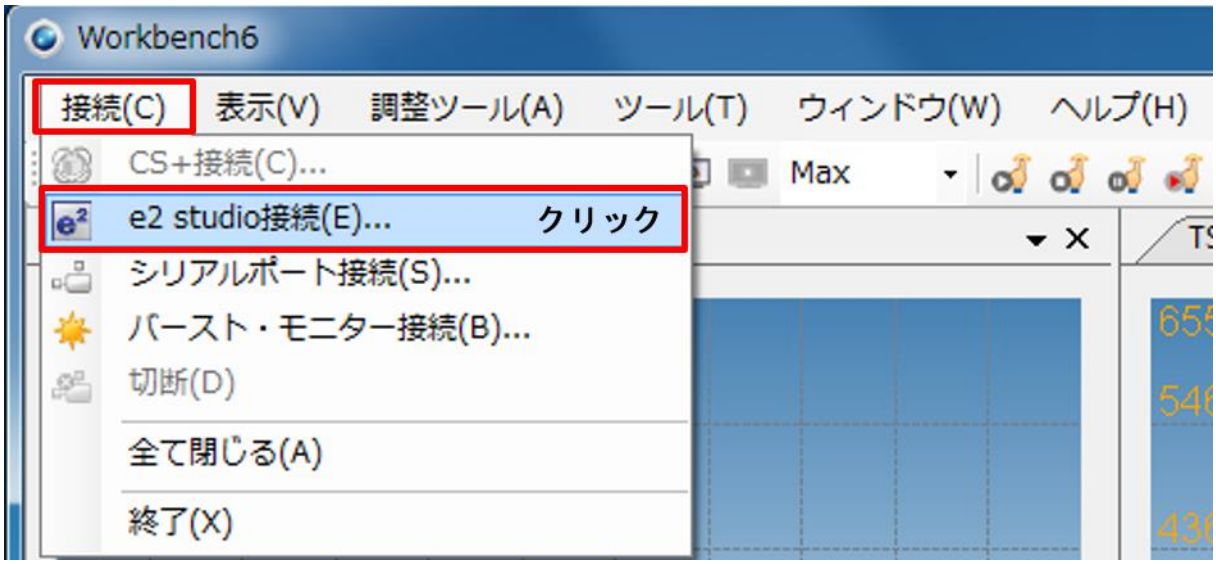
4 プロジェクトの実行

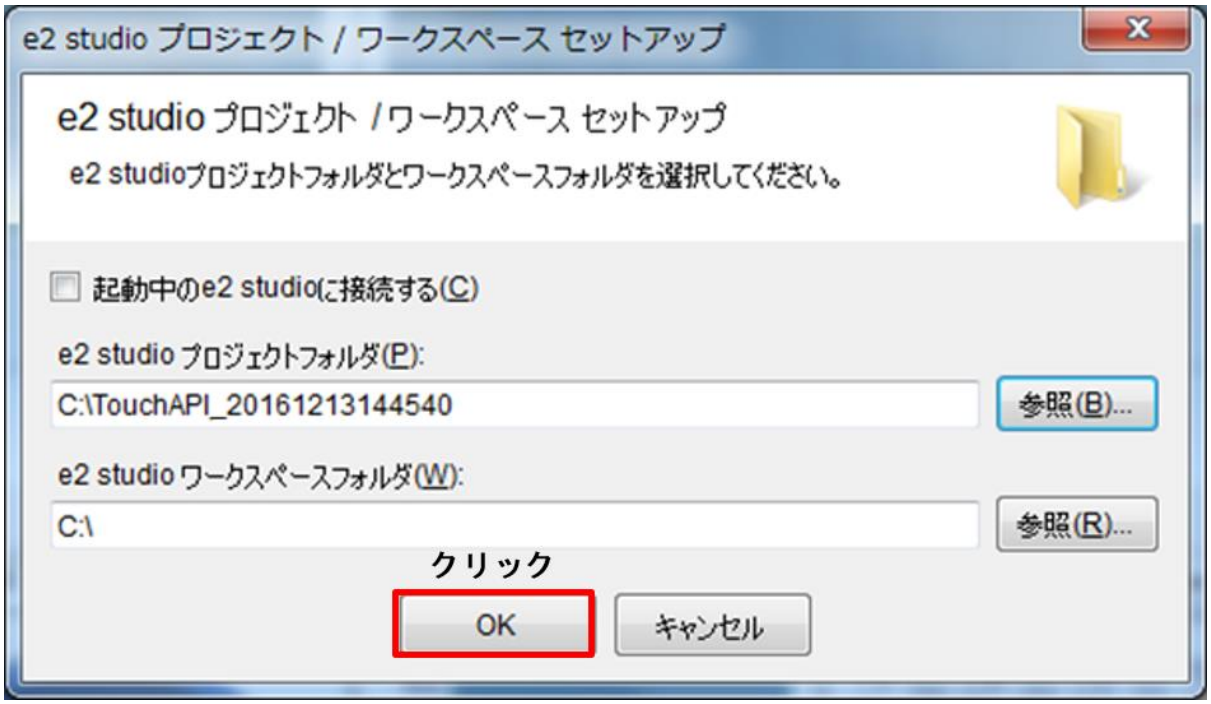


5個のボタンをそれぞれ押すと、対応するLEDが点灯します。近接センサーTS30から0.5～1cm程度に指を近づけるとLEDが点灯します。

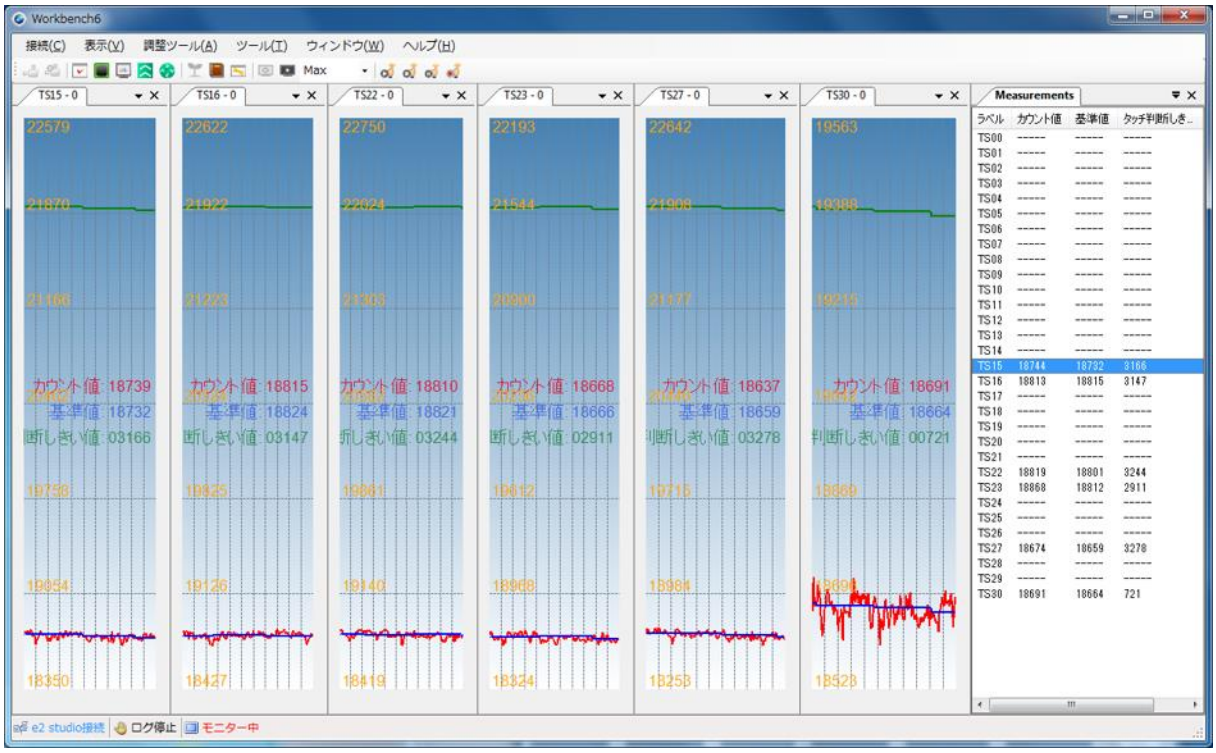
5 Workbench6のチューニング




Workbench6
接続(C)
表示(V)
調整ツール(A)
ツール(T)
ウィンドウ(W)
ヘルプ(H)
CS+接続(C)...
e2 studio接続(E)...
クリック
シリアルポート接続(S)...
バースト・モニター接続(B)...
切断(D)
全て閉じる(A)
終了(X)
Max


e2 studio プロジェクト / ワークスペース セットアップ
e2 studio プロジェクト / ワークスペース セットアップ
e2 studioプロジェクトフォルダとワークスペースフォルダを選択してください。
起動中のe2 studioに接続する(C)
e2 studio プロジェクトフォルダ(P):
C:\TouchAPI_20161213144540
参照(B)...
e2 studio ワークスペースフォルダ(W):
C:\
参照(R)...
クリック
OK
キャンセル


Workbench6
接続(C)
表示(V)
調整ツール(A)
ツール(T)
ウィンドウ(W)
ヘルプ(H)
Max
TS15 - 0
TS16 - 0
クリック
TS22 - 0



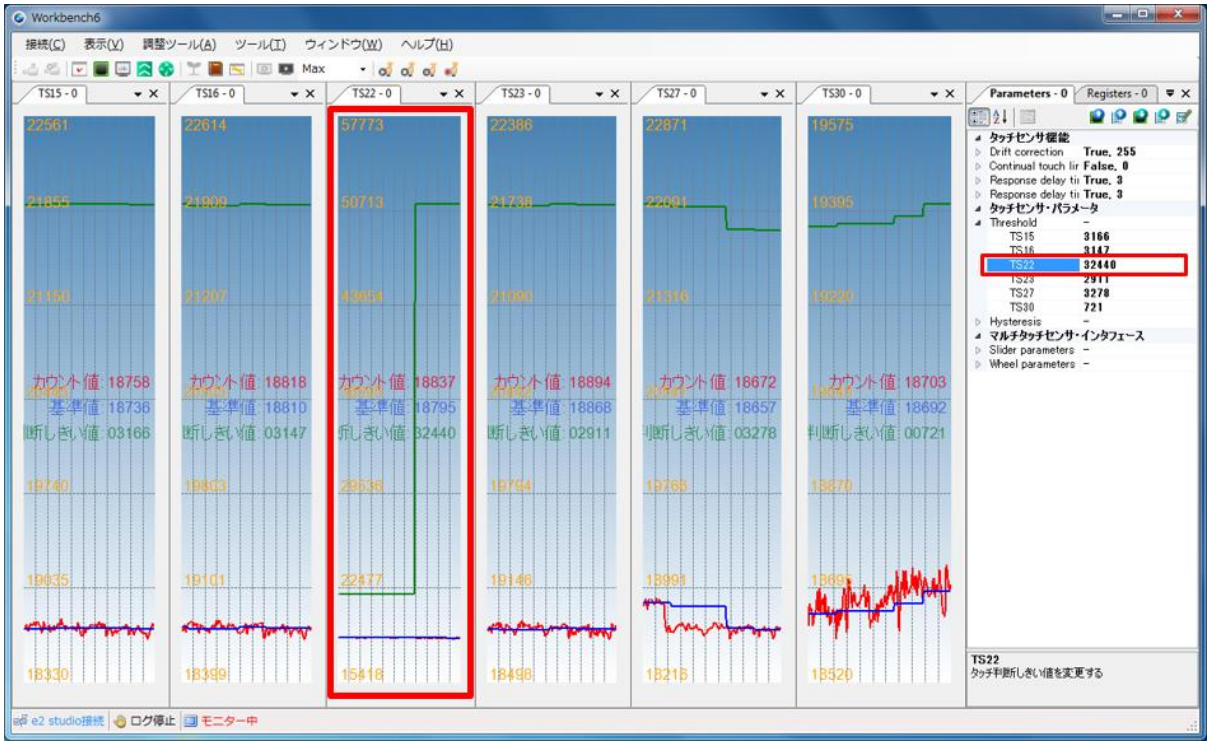
測定データは異なることもあります。

カウント値：断続的に変化するタッチ センサーの値

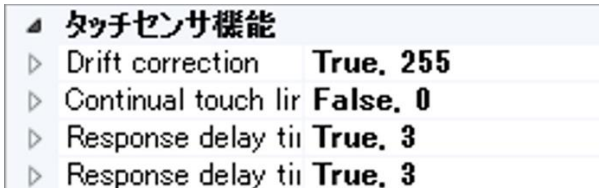
基準値：カウント値の平均値

判断しきい値：タッチ認識範囲値

Touch? = カウント値 > 基準値 + 判断しきい値



スレッショルド値は、次の通りパラメータ タブで変更可能です。



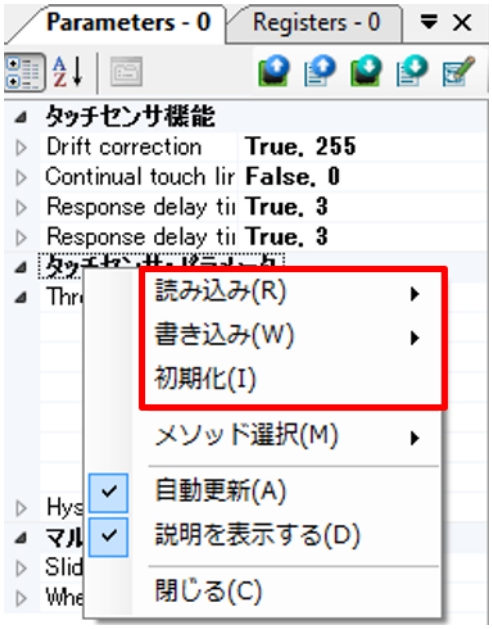
Touch sensor functionではReference Valueの計算速度、タッチ応答時間などを指定することができます。

Reference Valueは約4秒ごとにCount valueの値を平均して求めます。

Reference Valueの計算速度はDrift Correction部分を以下の式で補正することで速くすることができます。

≒ Drift Correction Value* 0.016 [Sec] (範囲:0 ～ 255)

タッチ応答時間はディフォルトでは3と設定されていますが、0に設定することを推奨します。



パラメータ タブで右クリックをすると、上のようなポップ アップ ウィンドウが開きます。

読み込み	ターゲットボードから読み込み	連結接続されたボードからパラメータ値を読んできます
	ファイルから読み込み	PCに保存されたパラメータ ファイルを読んできます
書き込み	ターゲットボードに書き込み	連結されたボードに保存します
	ファイルに書き込み	パラメータ ファイルに保存します
	TouchAPIに書き込み	プロジェクトに保存します
初期化		変更前の初期値で再設定します