

# ***DENSO***

Crafting the Core

## デンソークリエイト 1DAYインターンシップ マイクロマウス単体テスト手順書

オンライン

CONFIDENTIAL  
関係者外秘

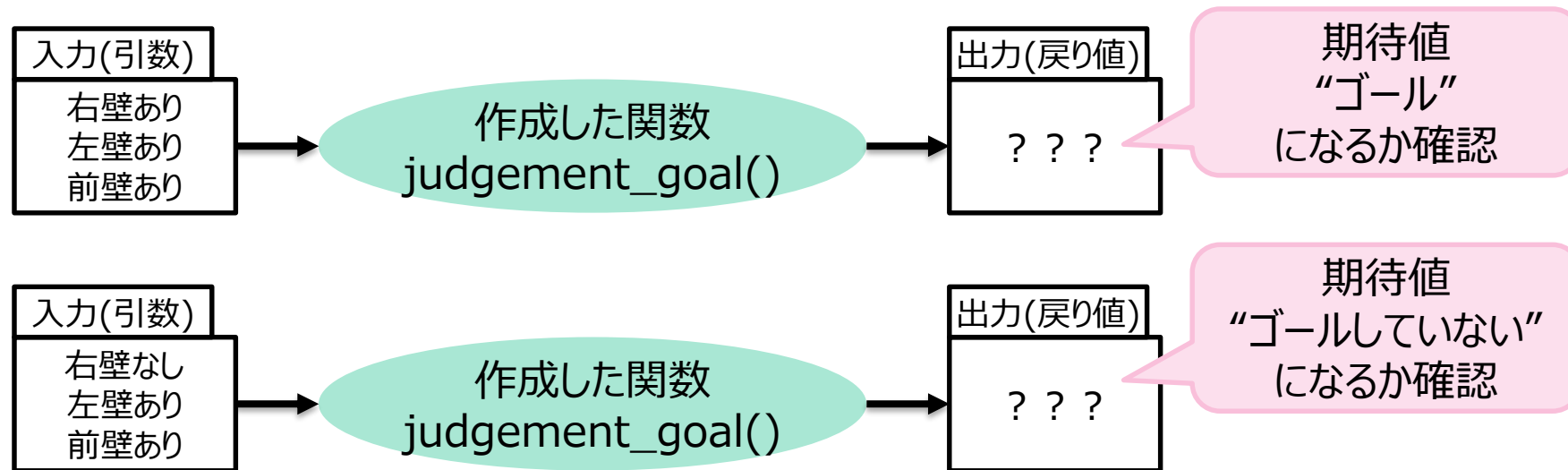
1. 単体テストの目的・概要
2. 単体テストの準備
  1. テスト用プロジェクトを開く
  2. テスト用プログラムをビルドする
  3. テスト用プログラムを動作させる
  4. テストケースを確認する
3. 単体テストを実施する

参考. 各種ウィンドウが消えてしまったときは

# 1. 単体テストの目的・概要

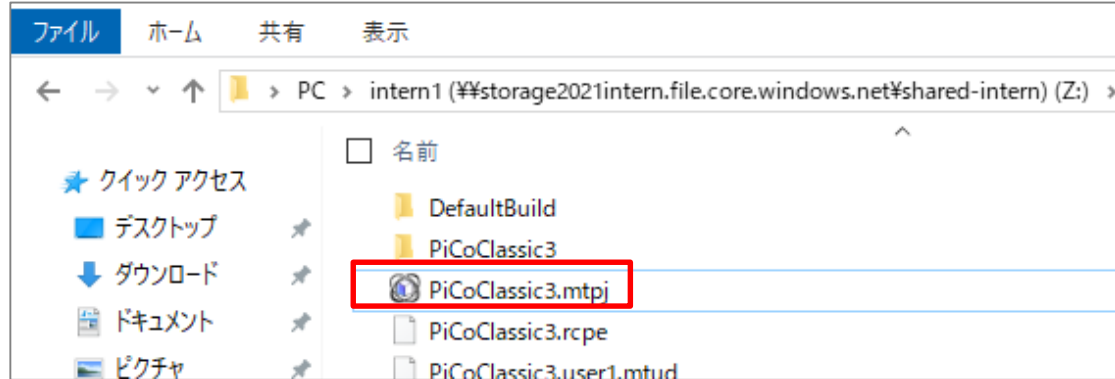
単体テストでは、作成した関数が設計通り作れているかどうかを確認します。

具体的には、テスト対象の関数(プログラム)をPC上で動作させ、入力に対して期待した出力が出てくるかどうかを確認します。

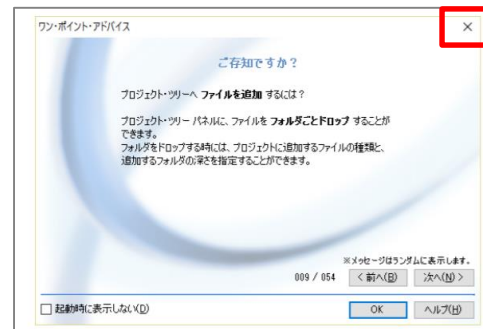
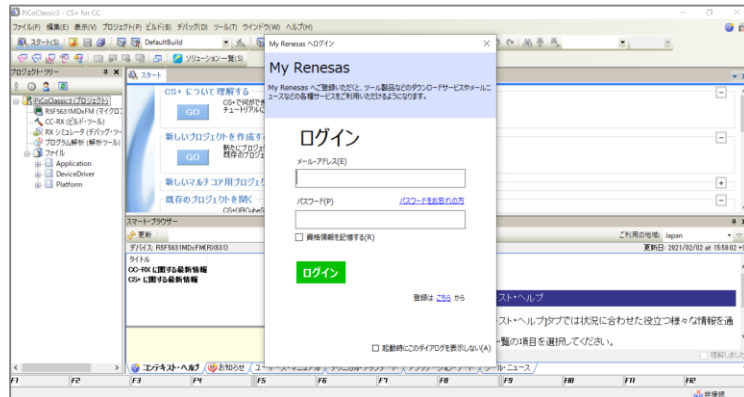


## 2-1. テスト用プロジェクトを開く (1/2)

**Z:¥2\_unit\_test** フォルダ内の **PiCoClassic3.mtpj** をダブルクリックしてください。  
開発時のプロジェクトとは別のファイルなので注意しましょう。



アプリケーションソフト “CS+ for CC” が起動します。  
起動時の画面はプログラム開発時とほぼ同じです。各種ダイアログが出たら、気にせず閉じてください。

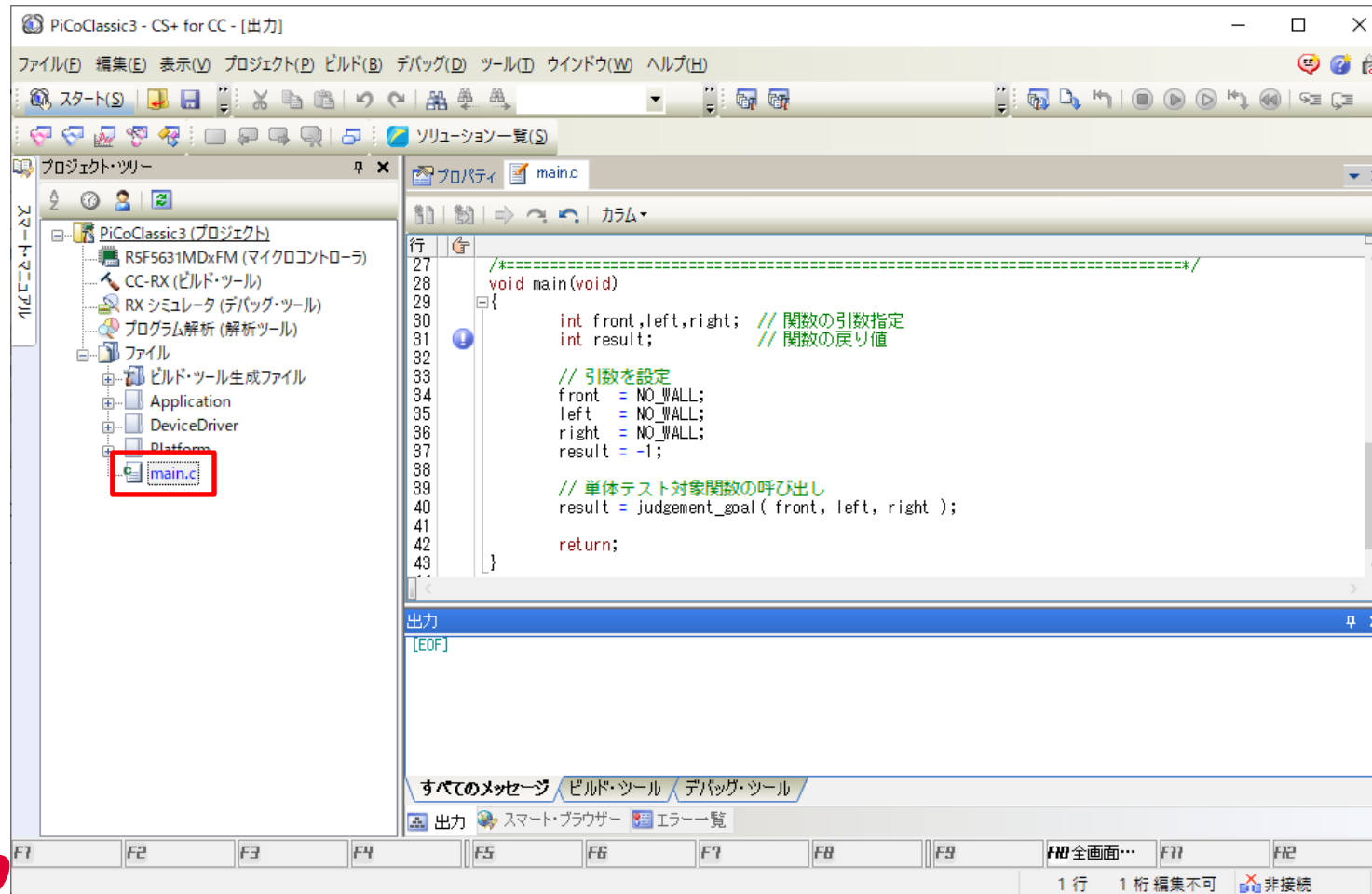


## 2-1. テスト用プロジェクトを開く (2/2)

プロジェクトを開くと、以下のような画面になります。

左側にある**プロジェクト・ツリー**から**main.c**をダブルクリックして開いてください。

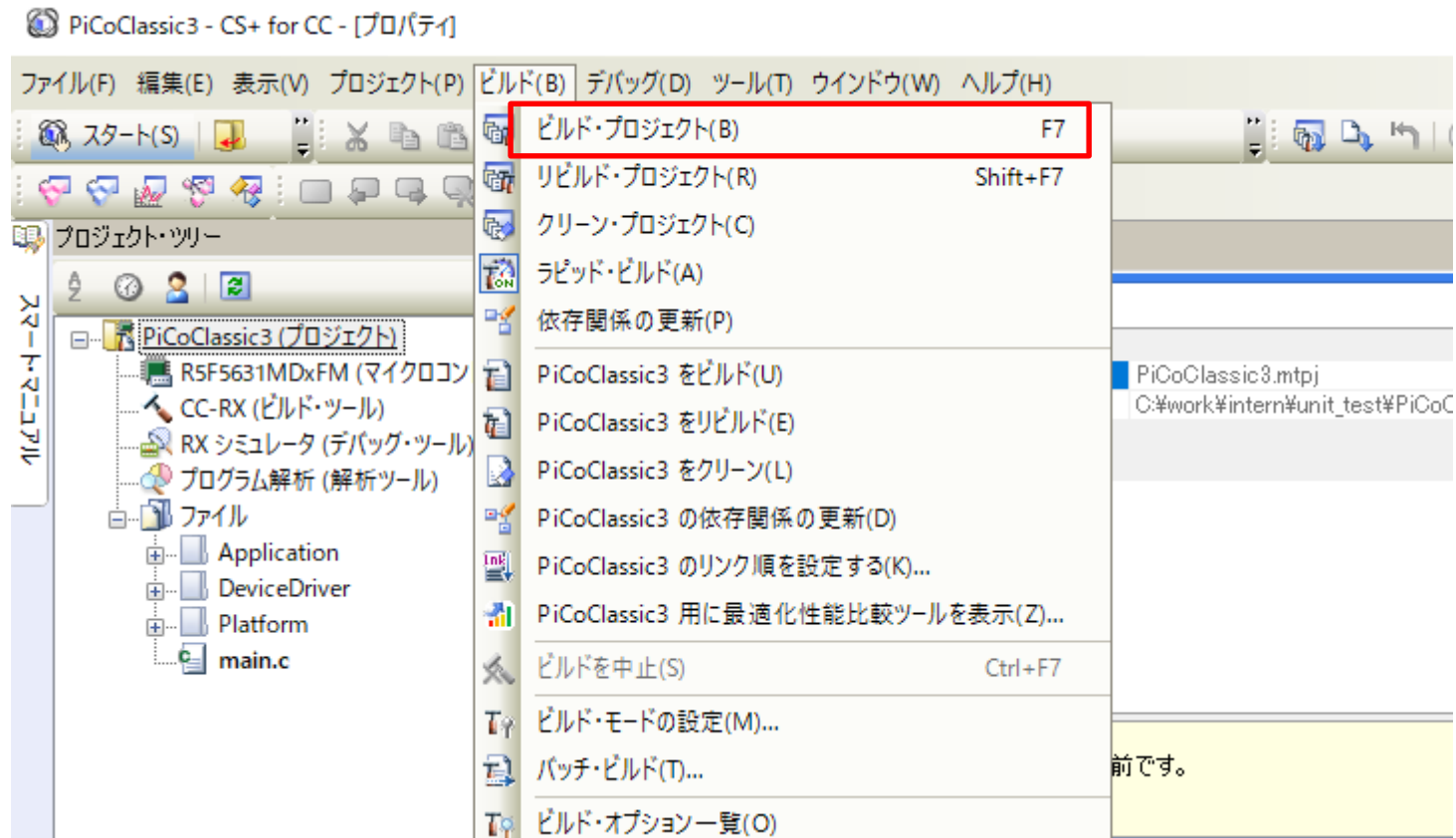
皆さんの作成した関数を呼び出すテスト用プログラムを用意してあります。



## 2-2. テスト用プログラムをビルドする (1/3)

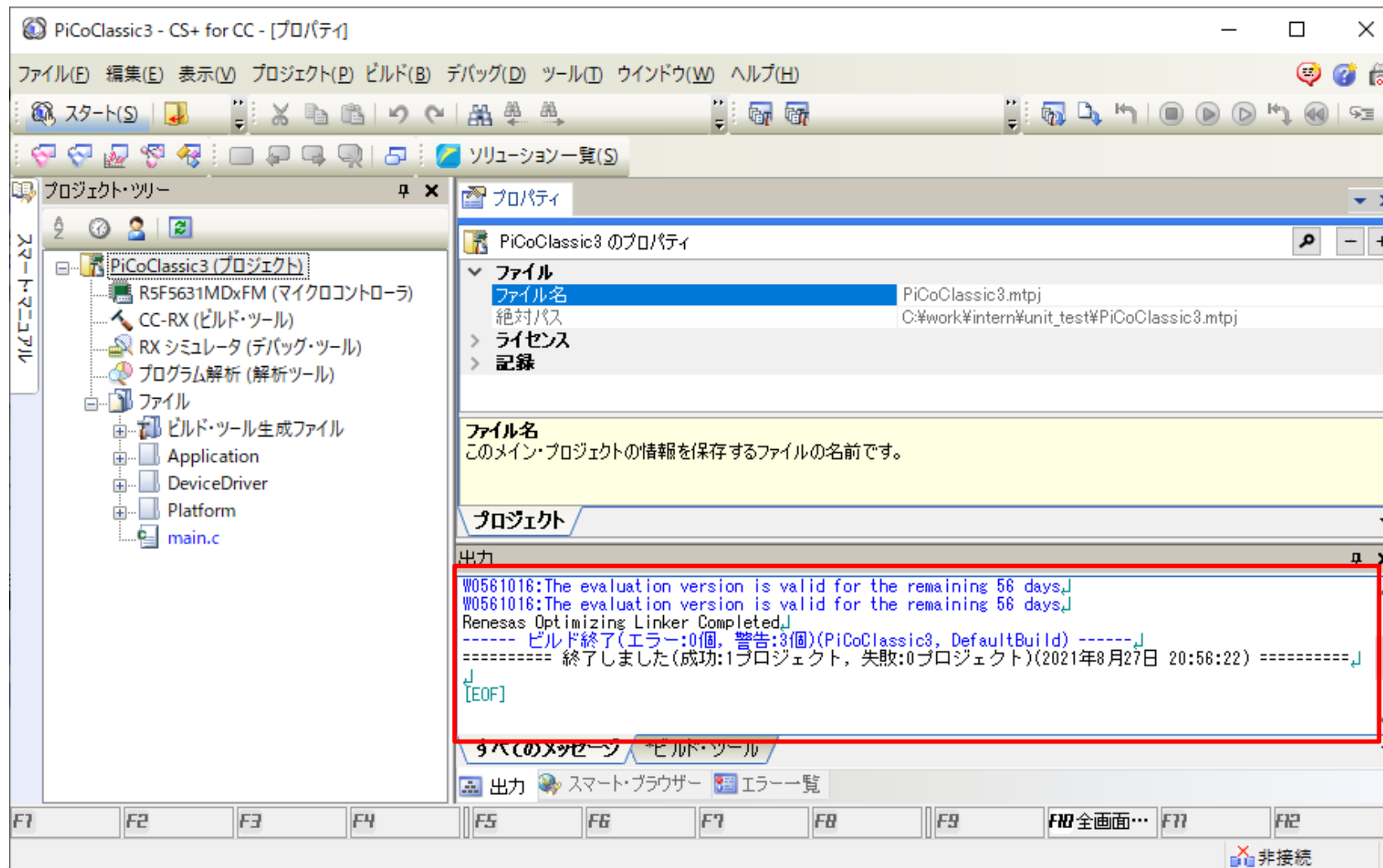
テスト用プログラムは変更する必要がないので、そのままビルドします。

メニューバーから「ビルド -> ビルド・プロジェクト(B)」を選択します。



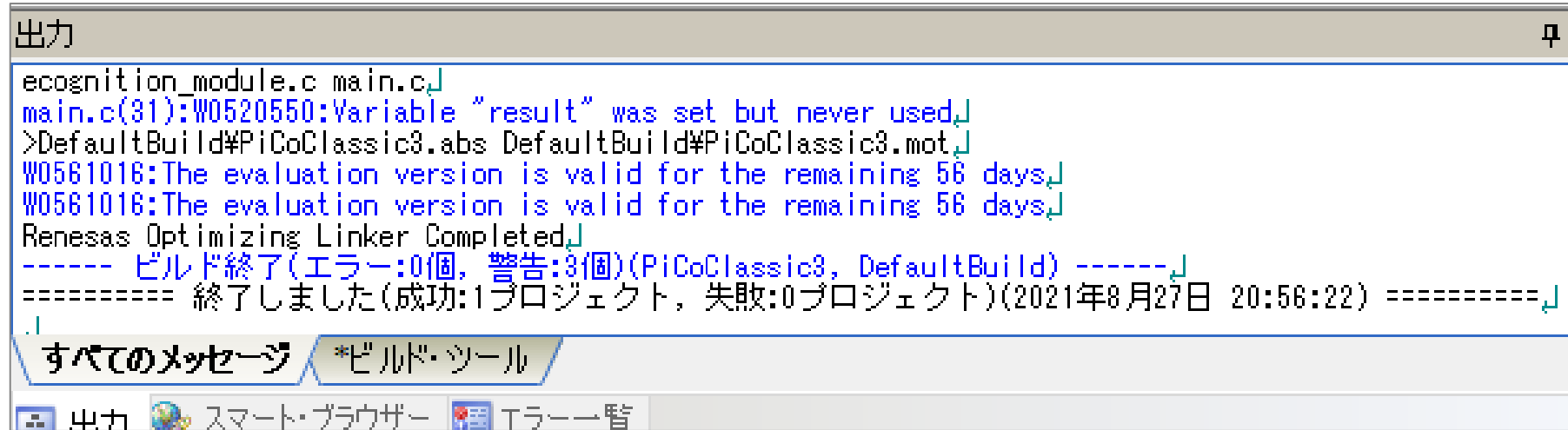
## 2-2. テスト用プログラムをビルドする (2/3)

ビルドを実行すると下記のような画面になります。



## 2-2. テスト用プログラムをビルドする (3/3)

画面の下のメッセージウィンドウの出力に次の文字が表示されたら、ビルドが成功です。



The screenshot shows the output window of an IDE. The title bar is '出力' (Output). The text inside the window is as follows:

```
ecognition_module.c main.c↓  
main.c(31):W0520550:Variable "result" was set but never used↓  
>DefaultBuild\PiCoClassic3.abs DefaultBuild\PiCoClassic3.mot↓  
W0561016:The evaluation version is valid for the remaining 56 days↓  
W0561016:The evaluation version is valid for the remaining 56 days↓  
Renesas Optimizing Linker Completed↓  
----- ビルド終了(エラー:0個, 警告:3個)(PiCoClassic3, DefaultBuild) -----↓  
===== 終了しました(成功:1プロジェクト, 失敗:0プロジェクト)(2021年8月27日 20:56:22) =====↓  
↓
```

Below the text, there are two tabs: 'すべてのメッセージ' (All Messages) and '\*ビルド・ツール' (Build Tools). The '\*ビルド・ツール' tab is selected. At the bottom of the window, there are three icons: a list icon, a globe icon labeled 'スマート・ブラウザー' (Smart Browser), and a list icon labeled 'エラー一覧' (Error List).

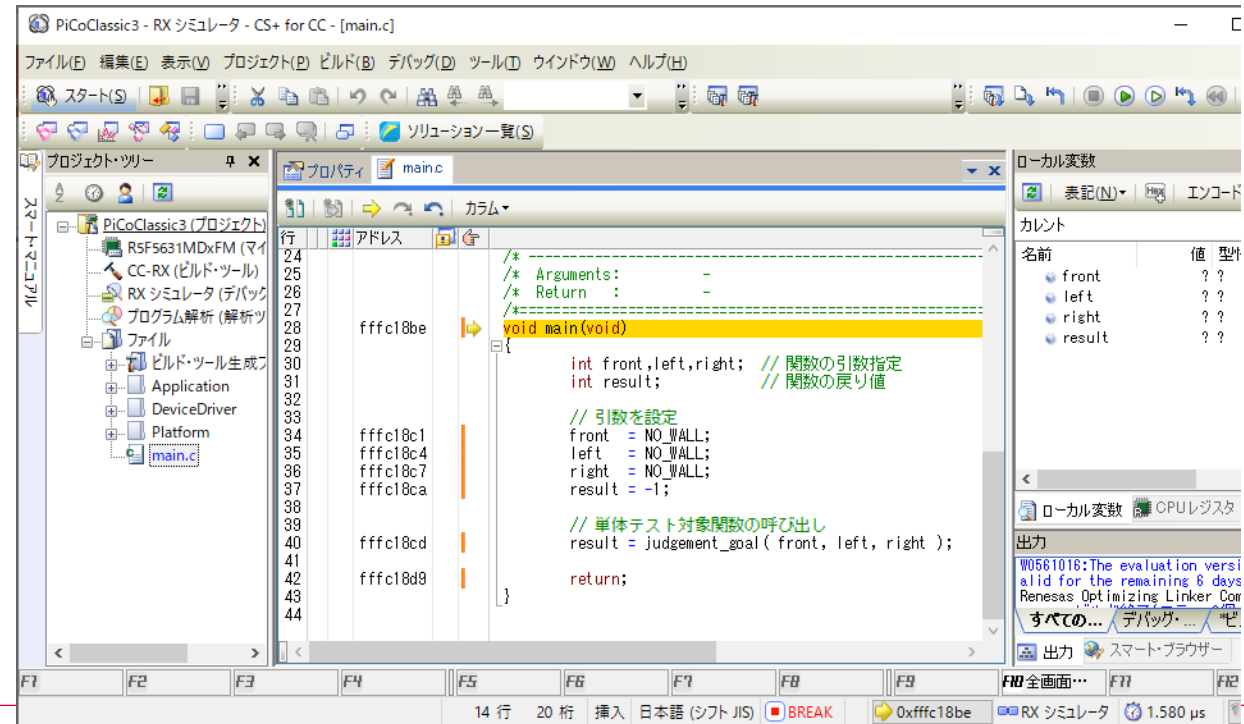
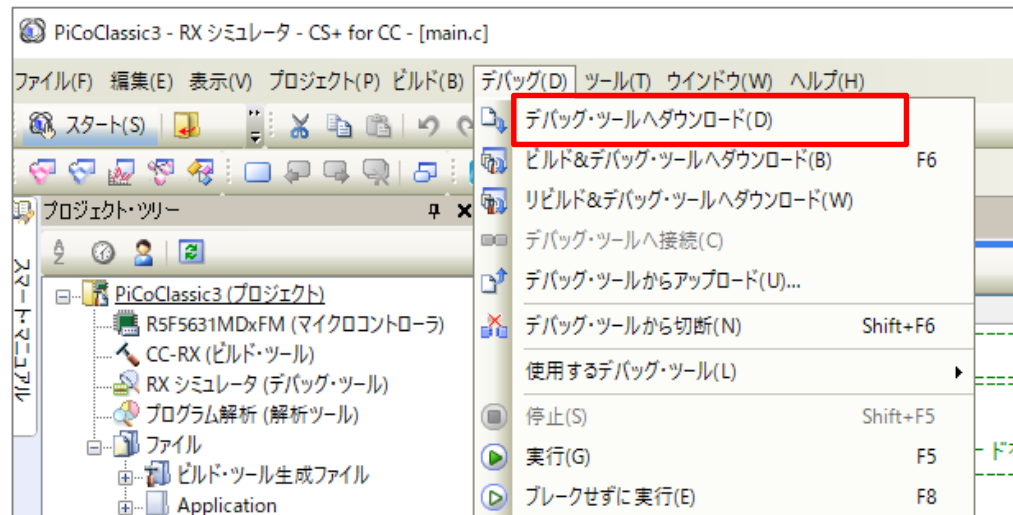
警告が合計3個出ますが問題はありません。



## 2-3. テスト用プログラムを動作させる (1/2)

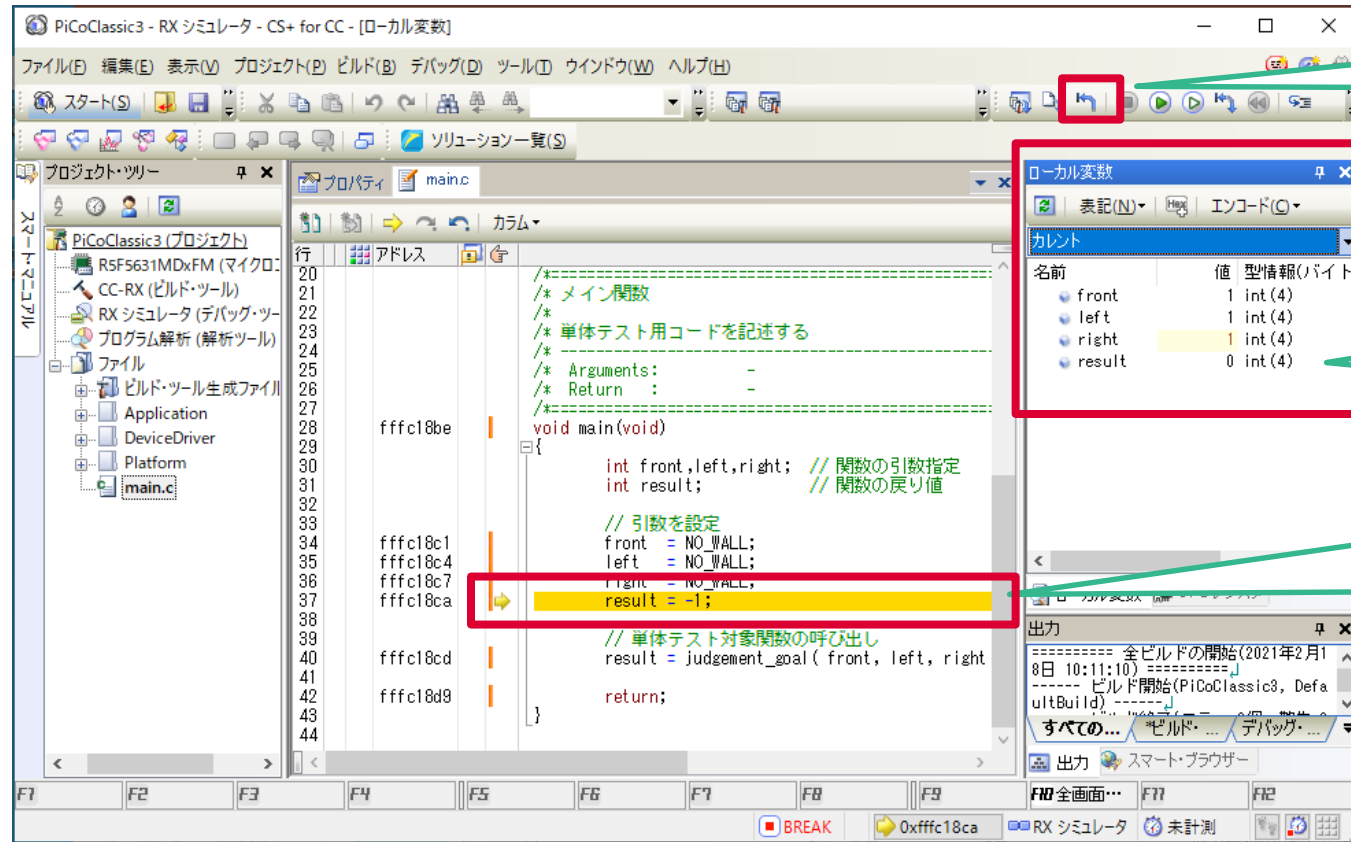
ビルドが成功したら、PC上でテスト用プログラムを動作させます。

メニューバーから「デバッグ -> デバッグ・ツールヘダownload(D)」を選択します。  
ダウンロードが完了すると以下のような画面になり、  
黄色い行（プログラム1行目）で一時停止した状態になります。



## 2-3. テスト用プログラムを動作させる (1/2)

これでPCでプログラムを動作させる準備ができました。  
動作中の画面は以下のようになっています。



CPUリセットで最初から実行しなおせる

変数の値が確認できる。  
また、数字をダブルクリックすると変更もできる。

黄色の行を実行する直前で止まっている。  
ステップ・インすると、その行を実行して次の行に移動する

## 2-4. テストケースを確認する

実施するテストの入力値・期待値のパターンは、設計書に記載してあります。  
設計書(Z:¥1\_設計書¥MicroMouse.iproj)を開き、**単体テスト**の項目を開きます。  
これから、右側に記載されているテストケースを順番に確認していくことになります。

MicroMouse

- システム要求
- ソフトウェア仕様
- ソフト構造
  - Application
    - 認知
    - 判断
    - 制御
  - Platform
  - DeviceDriver
  - アプリ・ランタイムデータ
  - 単体テスト**
  - ゴール判定
  - 外部テスト
  - データ型

キーワード

新規 参照 入力

追加できる要素がありません。

キーワード

MicroMouse ▶ 単体テスト ▶

単体テスト 単体テスト

テスト対象

1. テスト対象 ゴール判定

関数名

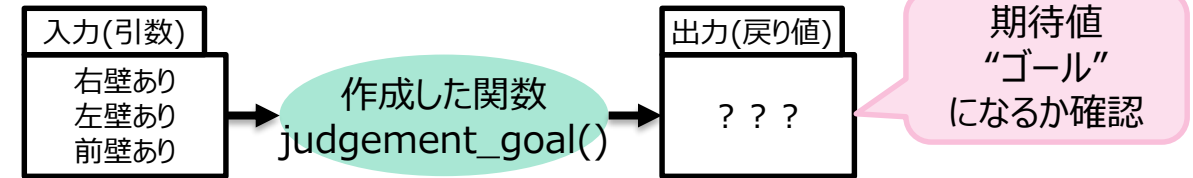
judgement\_goal

テストケース

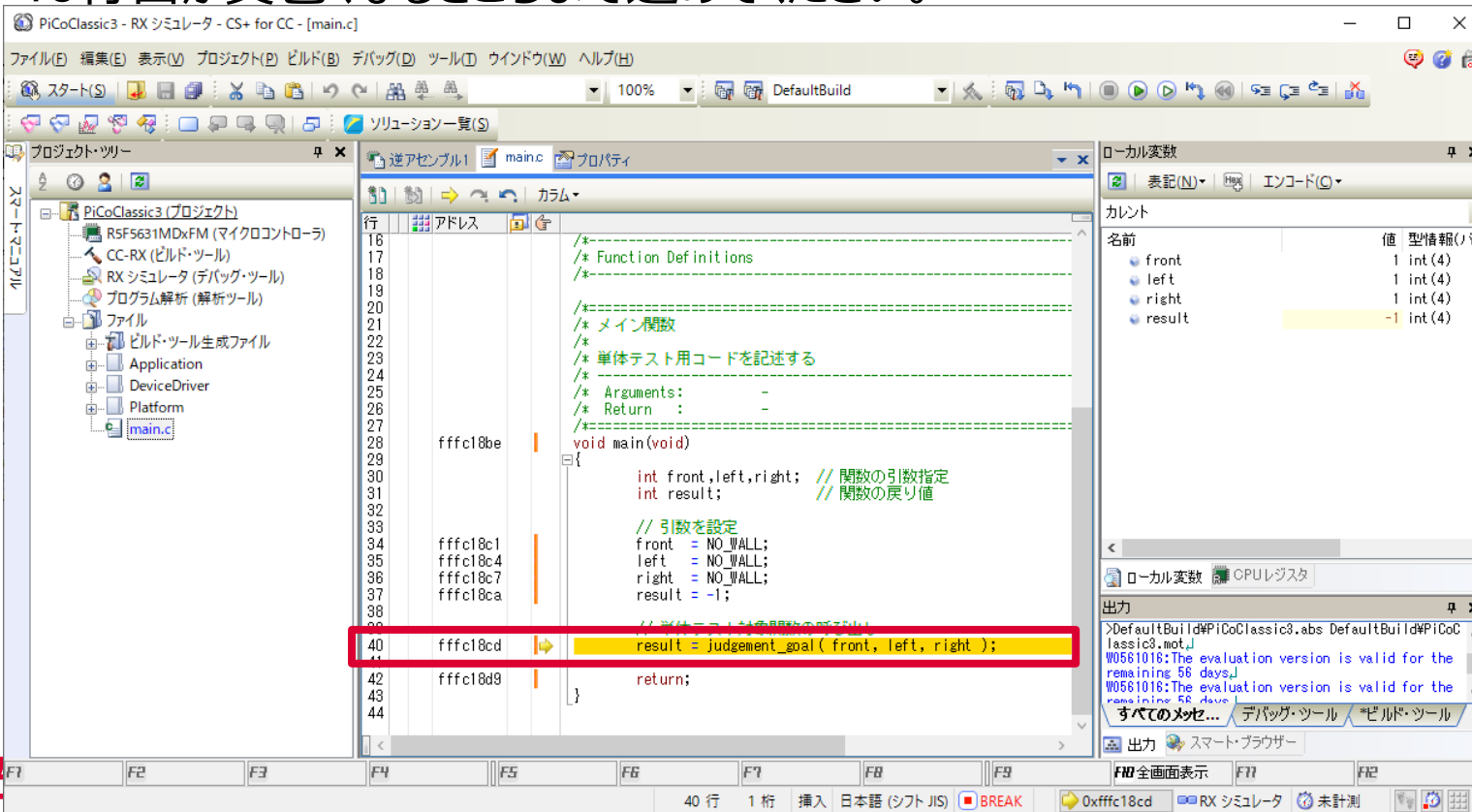
No.	テスト内容	入力値	期待値	合否
1.	3面壁ありの場合ゴールと判定する	existence_front_wall=WALL existence_left_wall=WALL existence_right_wall=WALL	goal_judgement_result=0	(未設定)
2.	前壁なしの場合ゴールではないと判定する	existence_front_wall=NO_WALL existence_left_wall=WALL existence_right_wall=WALL	goal_judgement_result=1	(未設定)
3.	左壁なしの場合ゴールではないと判定する	existence_front_wall=WALL existence_left_wall=NO_WALL existence_right_wall=WALL	goal_judgement_result=1	(未設定)
4.	右壁なしの場合	existence_front_wall=WALL	goal_judgement_result=1	(未設定)

### 3. 単体テストを実施する (1/4)

まずはテスト対象の関数が実行される直前までプログラムを進めます。

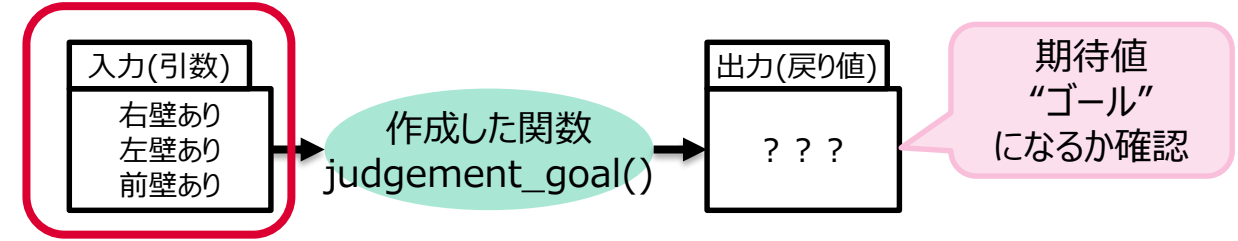


メニューバーの「デバッグ(D) -> ステップ・イン(I)」(ショートカットキーはF11)を用いて1行ずつコードを実行し、40行目が黄色くなるまで進めてください。



### 3. 単体テストを実施する (2/4)

テストケースの入力を設定します。



実施するテストケースの入力値に合わせて、「**ローカル変数**」にて対応する変数の値を変更してください。

テストケースの入力値

No.	テスト内容	入力値	期待値	合否
1.	3面壁ありの場合ゴールと判定する	existence_front_wall=WALL existence_left_wall=WALL existence_right_wall=WALL	goal_judgement_result=0	(未設定)
2.	前壁なしの場合ゴールではないと判定する	existence_front_wall=NO_WALL existence_left_wall=WALL existence_right_wall=WALL	goal_judgement_result=1	(未設定)
3.	左壁なしの場合ゴールではないと判定する	existence_front_wall=WALL existence_left_wall=NO_WALL existence_right_wall=WALL	goal_judgement_result=1	(未設定)
4.	右壁なしの場合ゴールではないと判定する	existence_front_wall=WALL existence_left_wall=WALL existence_right_wall=NO_WALL	goal_judgement_result=1	(未設定)

値をテストケースに応じて変更する

名前	値	型情報(バイト)
front	1	int(4)
left	1	int(4)
right	1	int(4)
result	0	int(4)

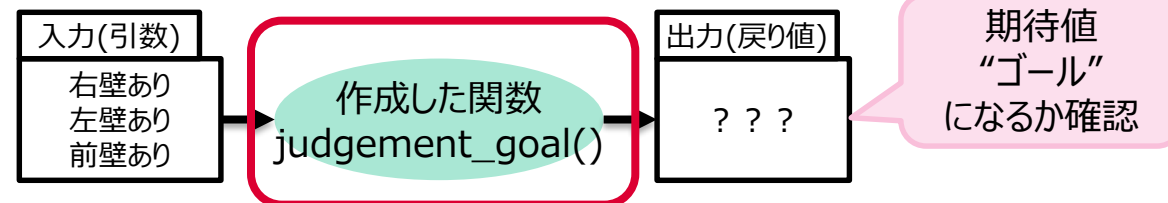
入力値(WALL/NO\_WALL)に対応する値(0/1)

説明	型への参照	値
前壁有無	int	NO_WALL(1):無 WALL(0):有

### 3. 単体テストを実施する (3/4)

CONFIDENTIAL  
関係者外秘

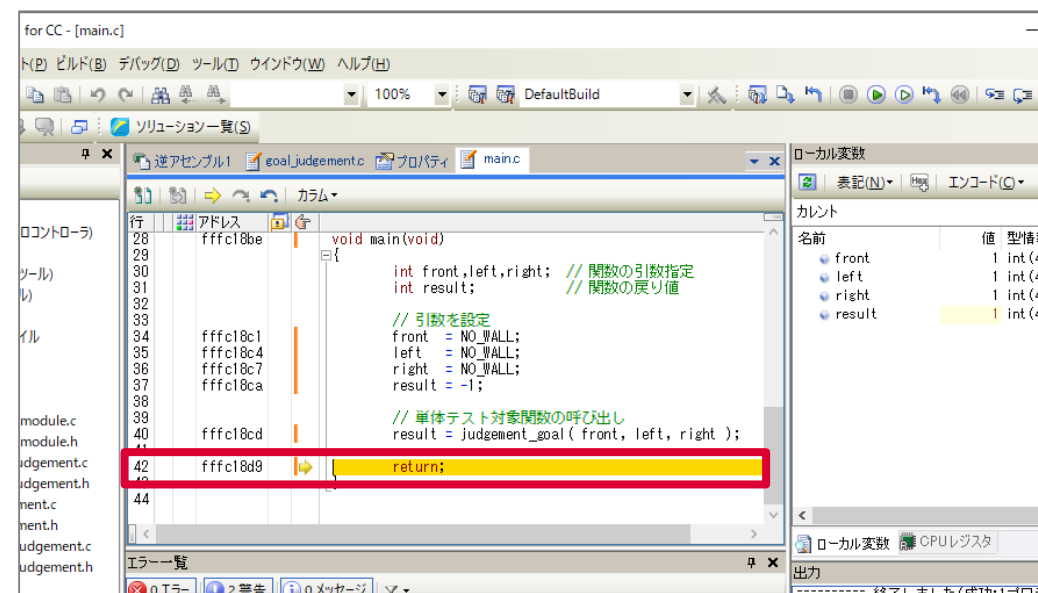
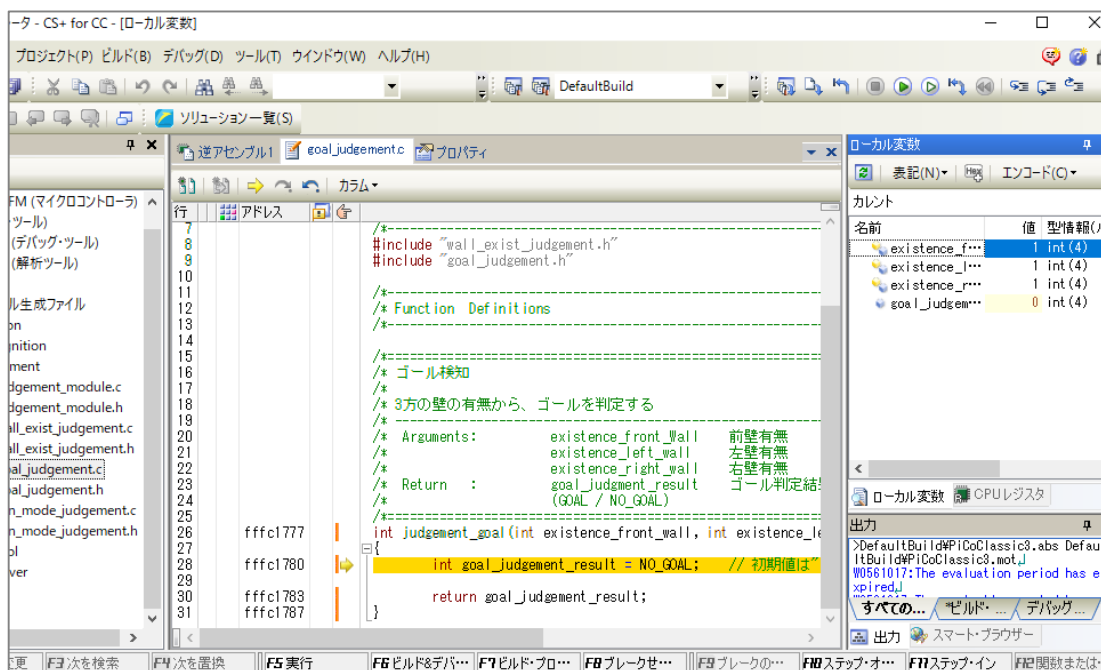
関数を動作させます。



再びステップ・インを用いて、コードを1行ずつ実行します。

皆さんが作成した関数(judgement\_goal)の中も1行ずつ実行し、想定通りの箇所を通っているか確認しましょう。

main関数の42行目が黄色くなるまで進めます。

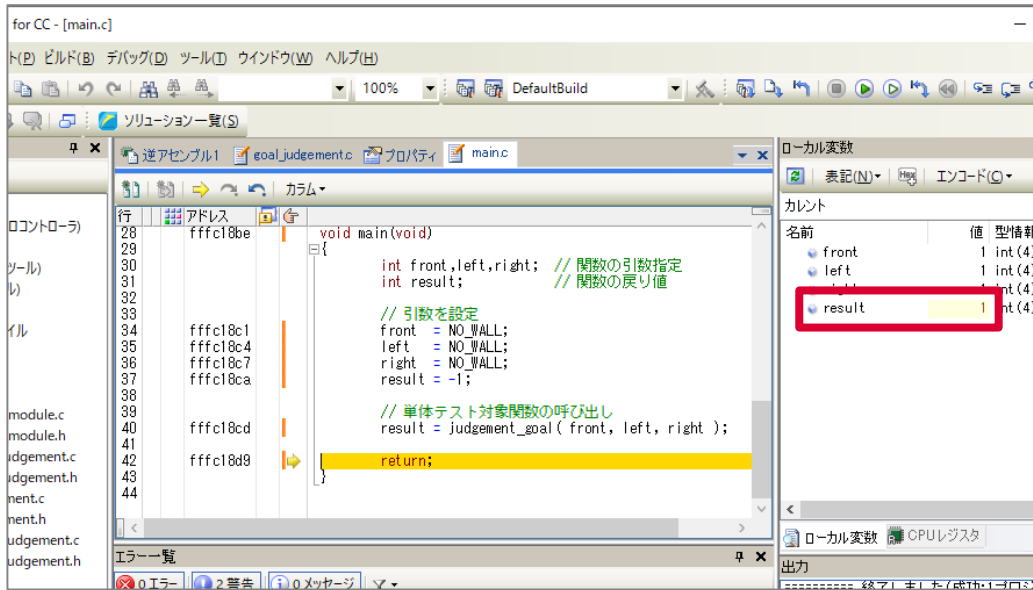
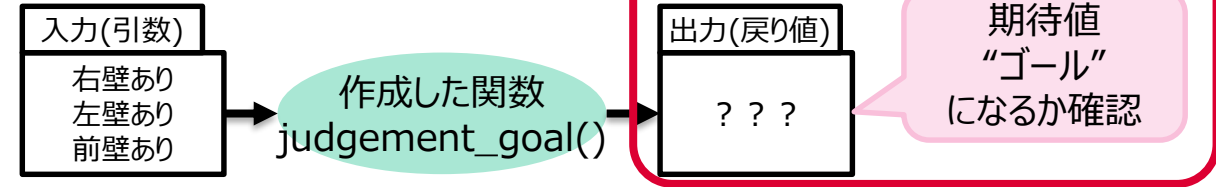




### 3. 単体テストを実施する (3/4)

関数の出力が期待値通りになっているか確認します。

「ローカル変数」の**result**の値が、  
テストケースの期待値と同じであることが確認できたら、  
そのテストケースは**合格**です。



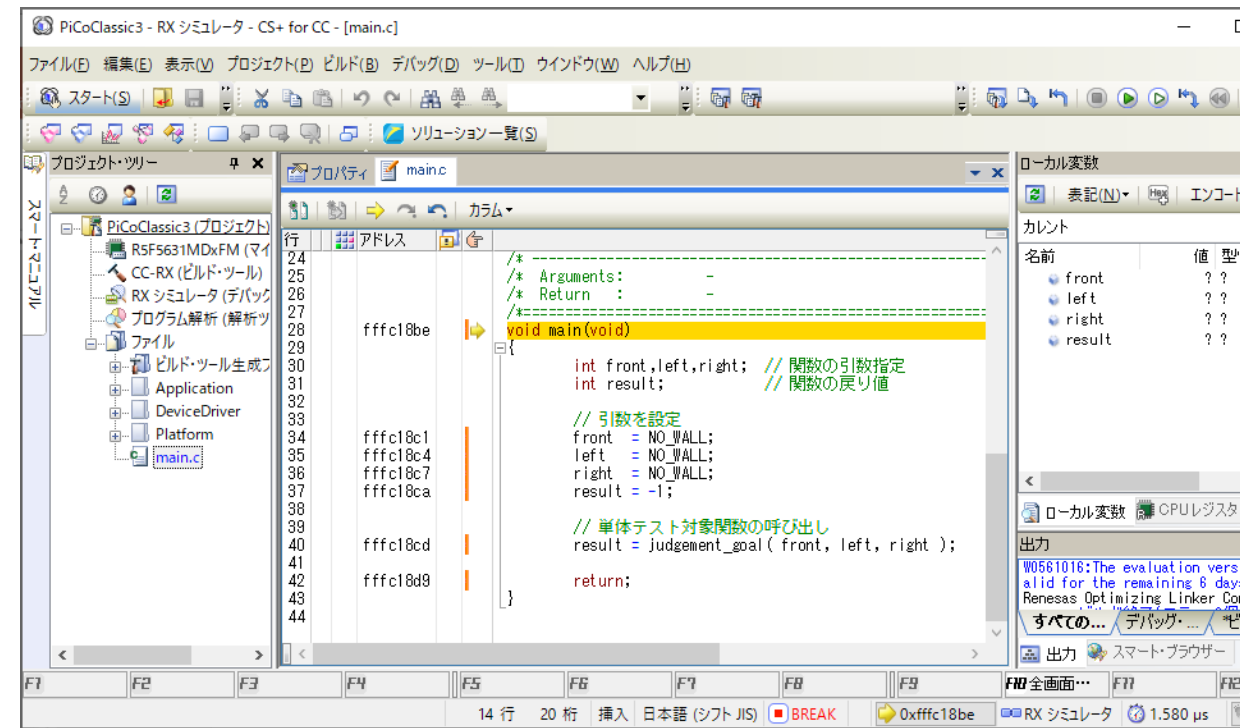
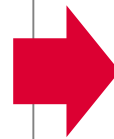
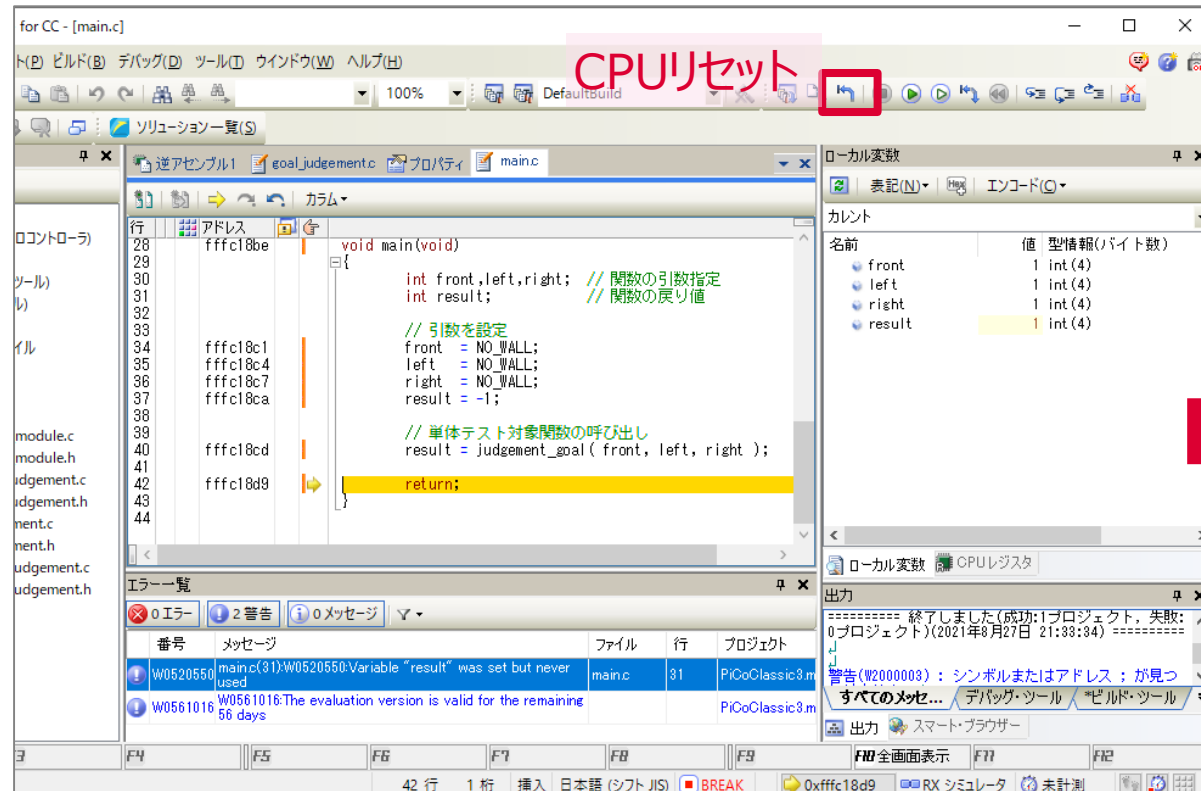
No.	テスト内容	入力値	期待値	可否
1.	3面壁ありの場合ゴールと判定する	existence_front_wall=WALL existence_left_wall=WALL existence_right_wall=WALL	goal_judgement_result=0	(未設定)
2.	前壁なしの場合ゴールではないと判定する	existence_front_wall=NO_WALL existence_left_wall=WALL existence_right_wall=WALL	goal_judgement_result=1	(未設定)
3.	左壁なしの場合ゴールではないと判定する	existence_front_wall=WALL existence_left_wall=NO_WALL existence_right_wall=WALL	goal_judgement_result=1	(未設定)
4.	右壁なしの場合ゴールではないと判定する	existence_front_wall=WALL existence_left_wall=WALL existence_right_wall=NO_WALL	goal_judgement_result=1	(未設定)
5.	前壁のみありの場合ゴールではないと判定する	existence_front_wall=WALL existence_left_wall=NO_WALL existence_right_wall=NO_WALL	goal_judgement_result=1	(未設定)
6.	左壁のみありの場合ゴールではないと判定する	existence_front_wall=NO_WALL existence_left_wall=WALL existence_right_wall=NO_WALL	goal_judgement_result=1	(未設定)
7.	右壁のみありの場合ゴールではないと判定する	existence_front_wall=NO_WALL existence_left_wall=NO_WALL existence_right_wall=WALL	goal_judgement_result=1	(未設定)
8.	3面壁なしの場合ゴールではないと判定する	existence_front_wall=NO_WALL existence_left_wall=NO_WALL existence_right_wall=NO_WALL	goal_judgement_result=1	(未設定)

### 3. 単体テストを実施する (4/4)

一つのテストケースが確認できたら、次のテストケースを確認するためにプログラムをリセットします。

「デバッグ(D) -> CPUリセット(T)」または、ツールバー上のCPUリセットボタンを押すと、プログラムはまた1行目に戻ります。

戻ったら、同じ手順で以降のテストケースを実施していきます。

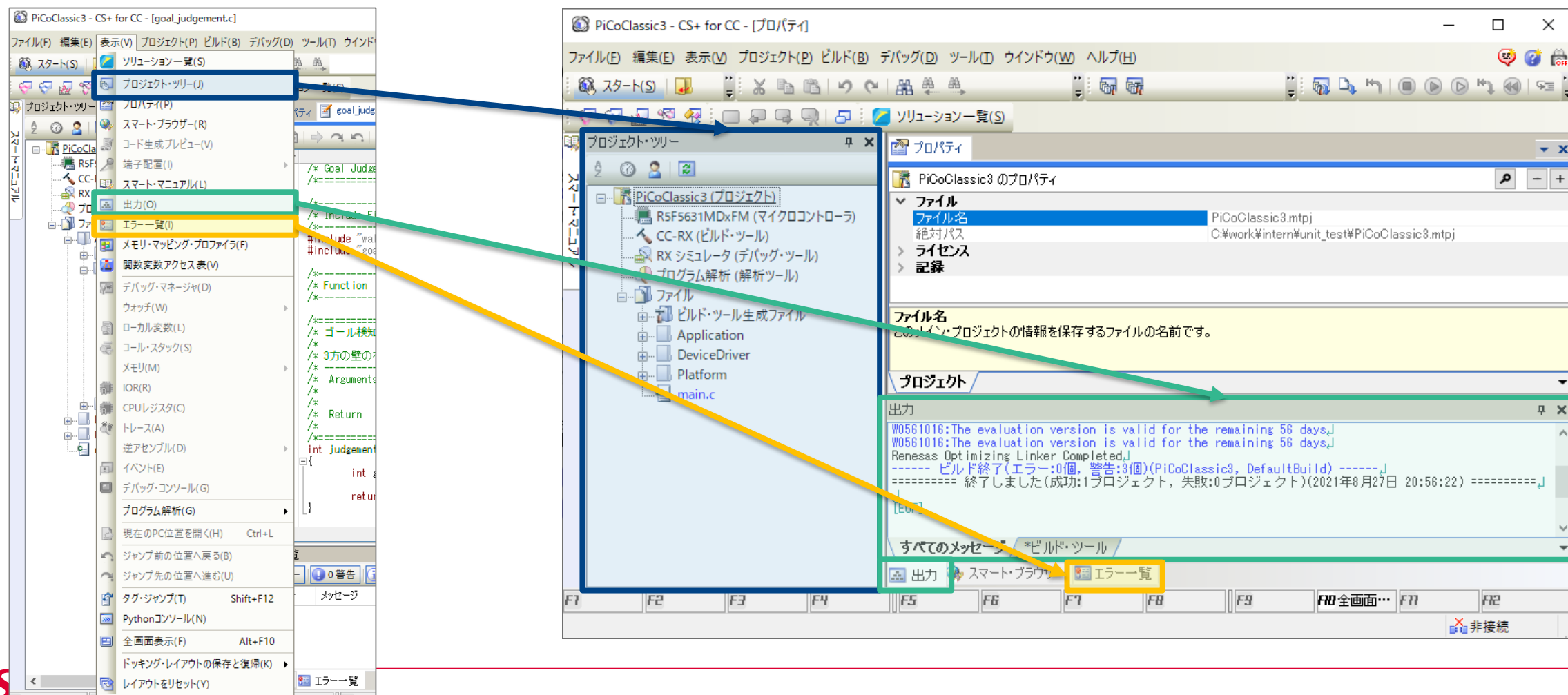




# 参考. 各種ウィンドウが消えてしまったときは (ソースコード作成時)

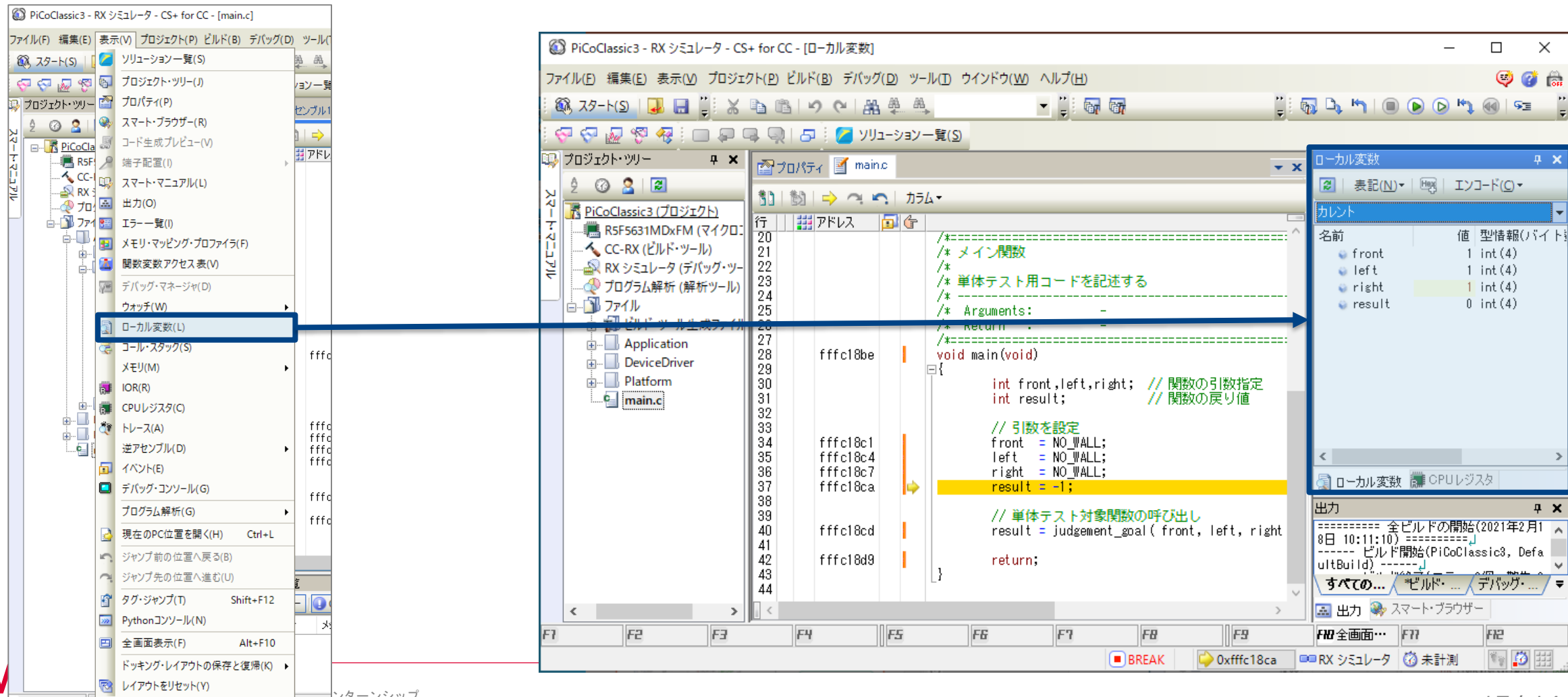
CONFIDENTIAL  
関係者外秘

ツール内の各種ウィンドウが消えてしまったときは、  
「表示(V)」メニューから表示したいウィンドウを選択すると再表示されます。



# 参考. 各種ウィンドウが消えてしまったときは（単体テスト実行時）

ツール内の各種ウィンドウが消えてしまったときは、  
「表示(V)」メニューから表示したいウィンドウを選択すると再表示されます。



***DENSO***

Crafting the Core