

# Simulink Library for MELSEC C 言語コントローラ リファレンスマニュアル



## 目次

履歴 .....	3
1. 概要 .....	4
2. 設定手順 .....	5
2.1 登録から使用までの流れ .....	5
2.2 S-Function 登録手順 .....	6
2.3 S-Function 使用手順 .....	8
2.3.1 作業フォルダへの C 言語ソースコピー .....	8
2.4 Embedded Coder でのコード生成 .....	10
2.5 CW Workbench でのビルド .....	11
3. S-Function プログラムについて .....	13
3.1 S-Function 仕様 .....	13
3.1.1 バスインターフェース S-Function .....	13
3.2 プログラムファイル概要 .....	14
添付資料 コンフィギュレーションパラメータ設定例 .....	15



## 履歴

バージョン	内容	日付
Ver. 1.00A	新規作成	2015.03.01



# 1. 概要

本書は、MATLAB®/Simulink®にて、C 言語コントローラ専用関数の機能を持った、S-Function の Function Block プログラム使用に関する説明書です。

本書記載のプログラムを使用するために必要なデータは以下となります。

ファイル名	概要
CCPULib.slx	C 言語コントローラインターフェースの Simulink ライブラリです。
slblocks.m	CCPULib.slx のライブラリを、Simulink ライブラリブラウザーに表示するために必要なファイルです。
src.zip	Simulink ライブラリの処理が記述されたプログラムファイルです。 *.c と *.h のファイルで構成されており、一つのライブラリに対し一つずつプログラムファイルがあります。
MathWorks_Sim_Q24DHCCPU_sample_j.pdf	本文書です。プログラム、設定方法の概要を記載しています。

本書で解説するプログラムに使用するソフトウェアは以下となります。

ソフトウェア名(形名)	説明
CW Workbench (SW1DND-CWWLQ24-E)	プログラムの作成、デバッグを行います。

本書で使用する MathWorks 製品は以下となります。

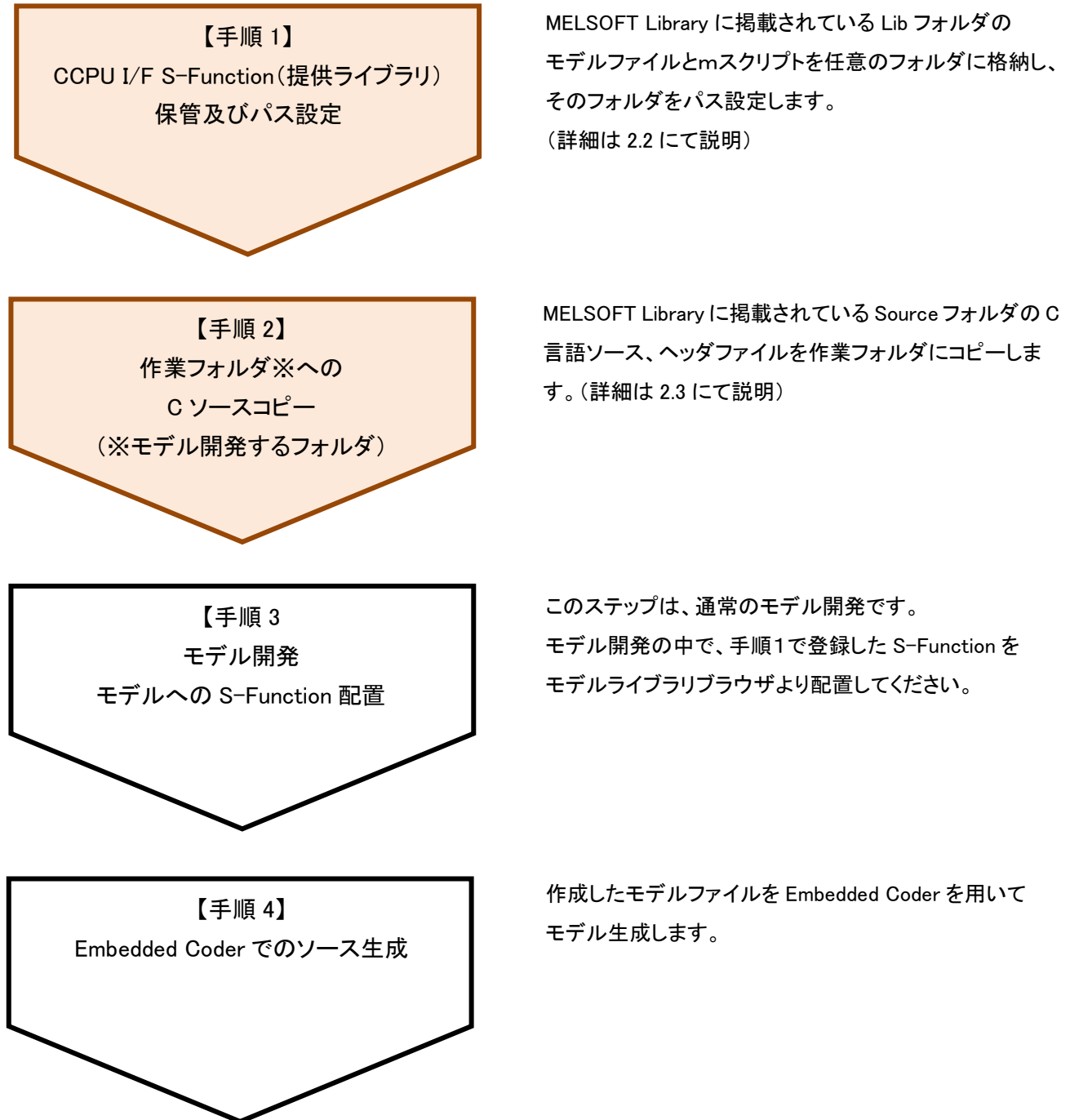
製品名	バージョン
MATLAB	R2014a
Simulink	R2014a
Embedded Coder	R2014a



## 2. 設定手順

### 2.1 登録から使用までの流れ

C 言語コントローラの I/F 用 S-Function を MATLAB/Simulink 上で配置し、ソースを生成するまでの設定手順の流れを下図のフロー図にて示します。



## 2.2 S-Function 登録手順

S-Function を登録し、ライブラリブラウザーへ表示します。

### 1) PC にモデルファイルのコピー

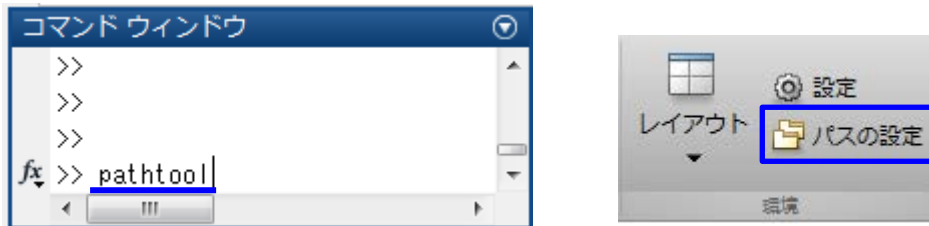
MELSOFT Library に掲載されている Lib フォルダを任意のフォルダに格納します。

※ここでは「C:\Program Files\MATLAB」以下に“Lib” フォルダをコピーしたこととし、手順を進めていきます。

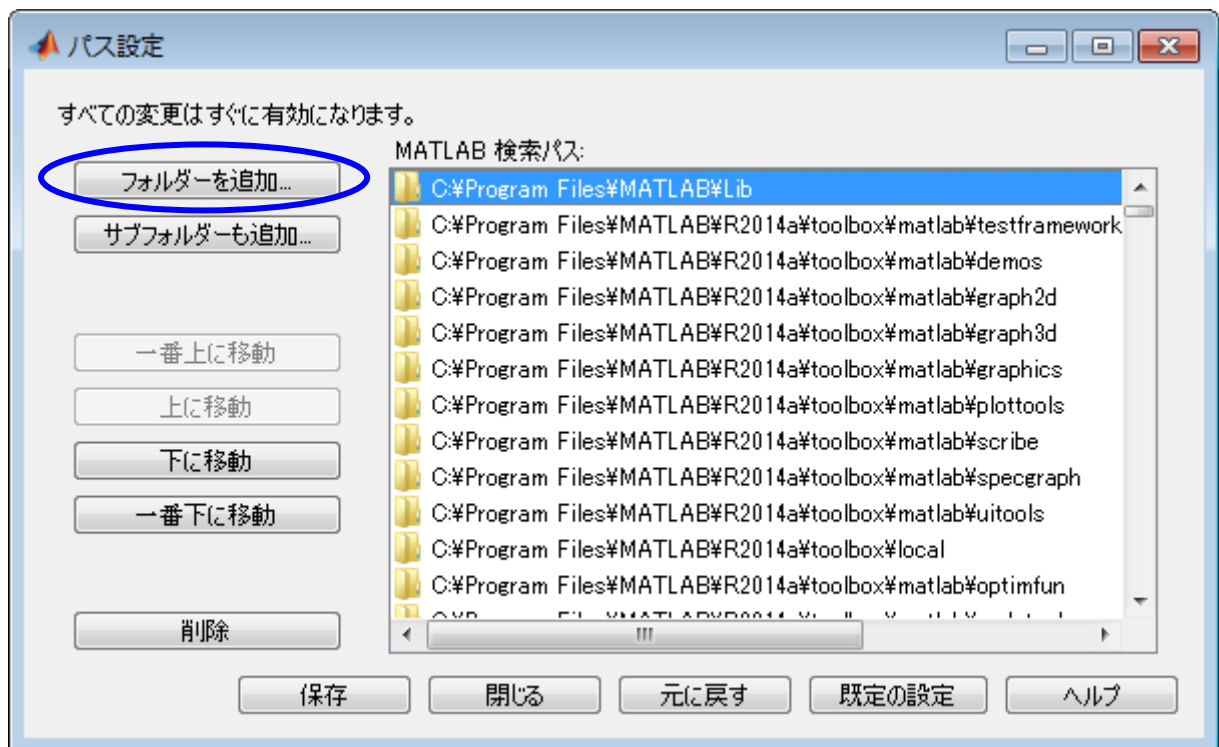
### 2) パス設定

MATLAB のパス設定に、1) で設定したフォルダを追加します。

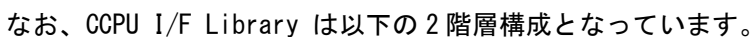
MATLAB コマンドウィンドウで“pathtool”と入力または、MATLAB ホームタブの「パスの設定」をクリックします。



下図ウィンドウが表示されるので「フォルダーを追加...」ボタンをクリックし、1) で格納したフォルダを追加します。



モデル開発にて、「ライブラリブラウザー」をクリックし、Simulink ライブラリブラウザーを表示します。ライブラリ一覧に“CCPU I/F Library”が追加されています。この追加ライブラリより S-Function を配置し、モデル開発を行ってください。



```
CCPU I/F Library — BUS I/F Library — sfBUS_In_BitEx
                                     | sfBUS_Out_BitEx
                                     | sfBUS_ReadDevice
                                     | sfBUS_WriteDevice
                                     | sfBUS_FromBuf
                                     | sfBUS_ToBuf
```

## 2.3 S-Function 使用手順

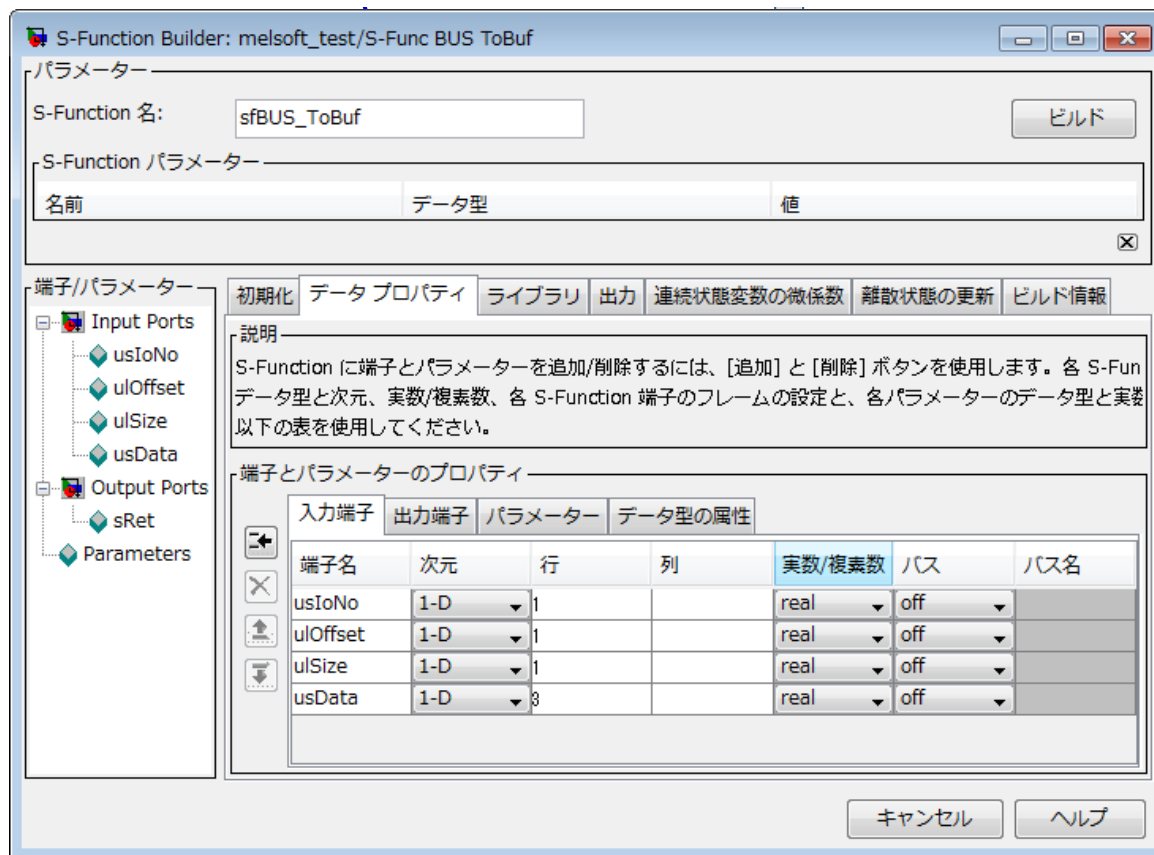
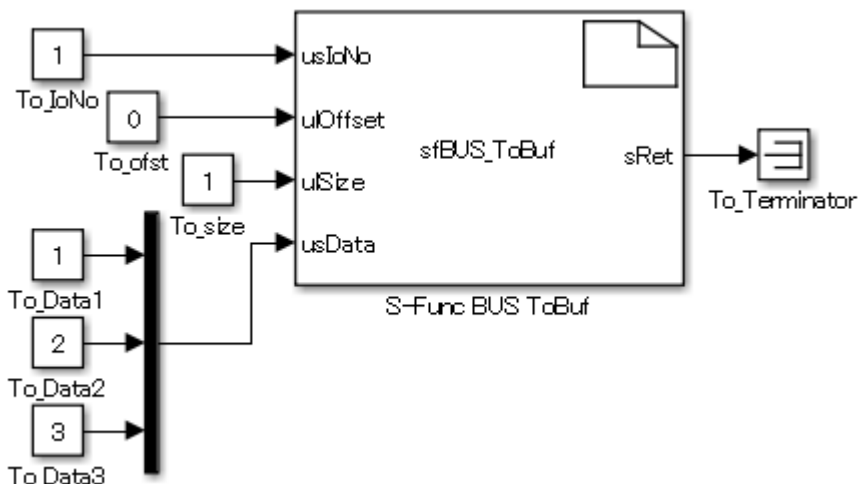
### 2.3.1 作業フォルダへのC言語ソースコピー

S-Function の呼び出し元のC言語ソース・インクルードファイルを、作業フォルダ(モデル作成のカレントフォルダ)へコピーします。※呼び出し元ファイルの確認は、各 S-Function の「ライブラリ」からご確認ください。

#### S-Function のパラメータ修正ビルド

S-Function に接続される端子の配列変更を行い、S-Function をビルドします。

ここでは、簡単な例を挙げて配列変更及びビルドの手順を説明します。



3




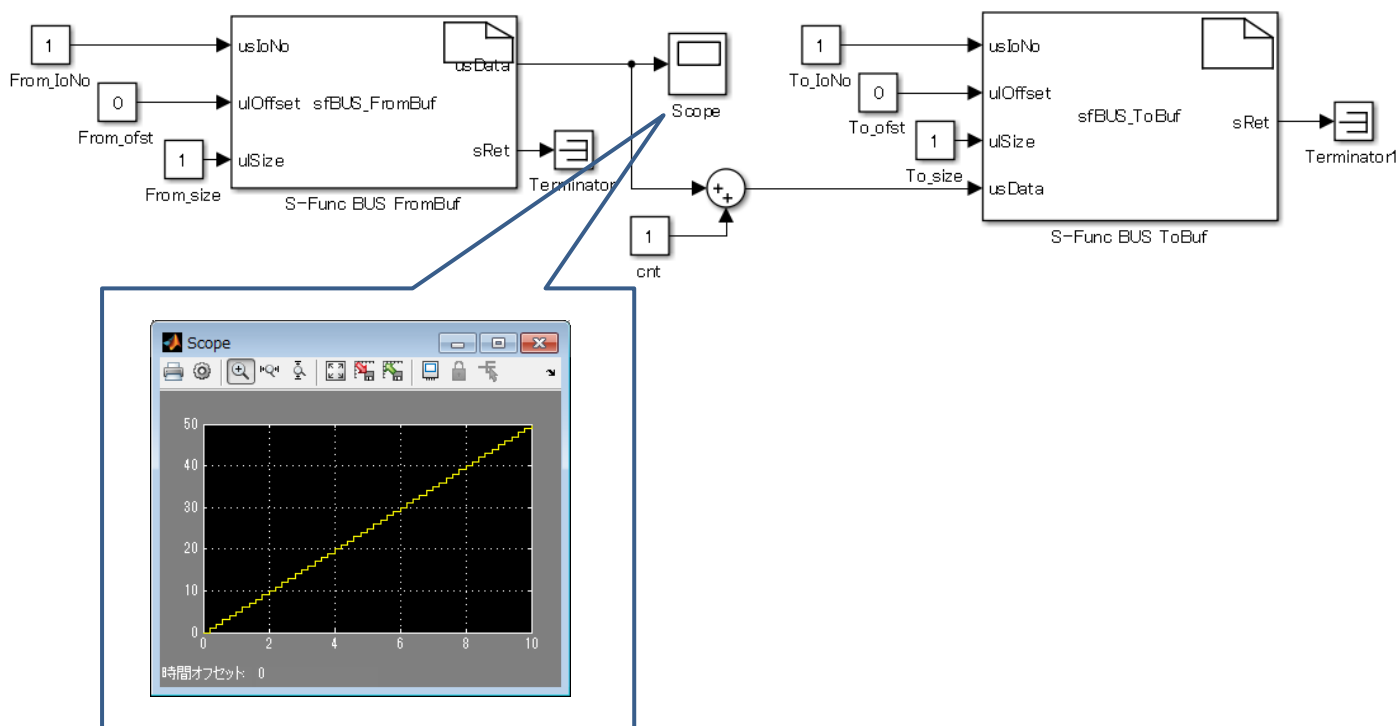
モデルに配置した S-Function をダブルクリック（図①）し、行数が可変の端子について、配列数と同じ行数となるように修正します。（図②）※同じ配列数の場合は、変更不要です。

S-Function をビルドします。（上図③）

### 【注意事項】

- ・ 配列数などパラメータを変更していない場合も、必ずビルドしてください。
- ・ モデルに配置した S-Function は全てビルドしてください。

ビルドが完了すると、 再生ボタンからシミュレーションが行えるようになります。



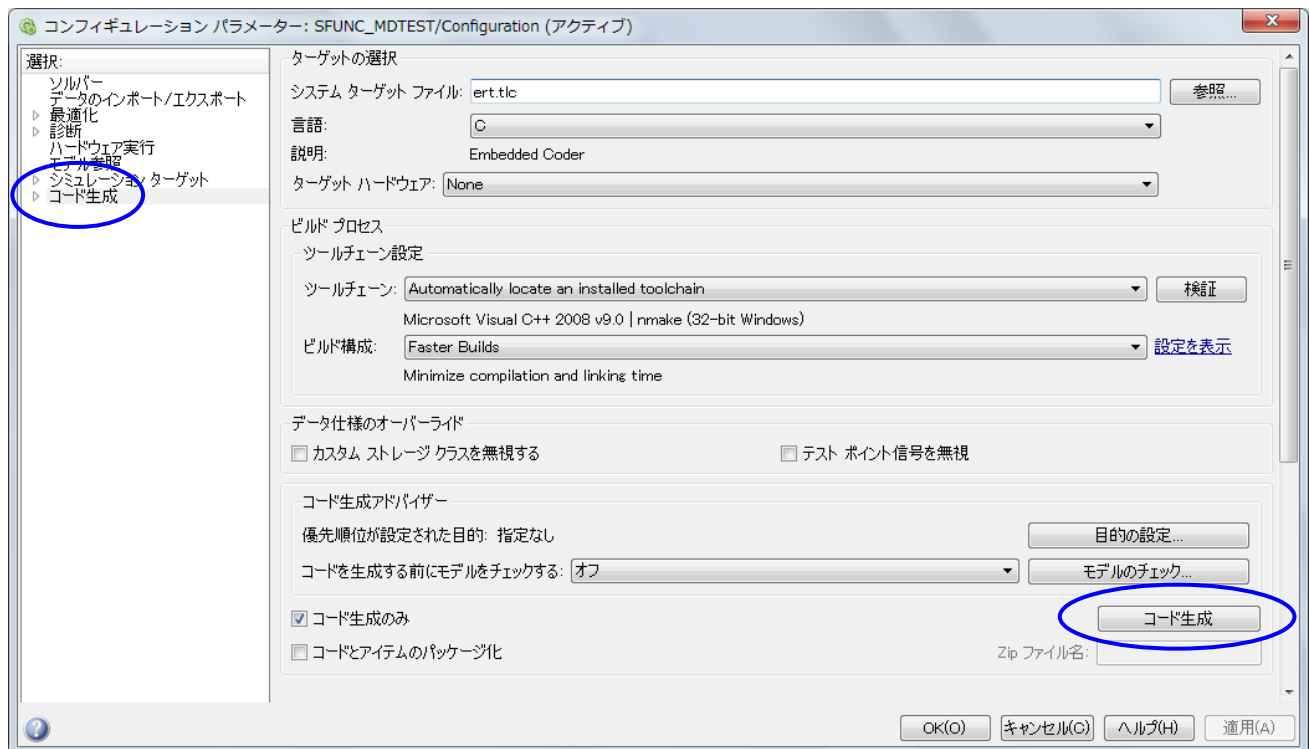
## 2.4 Embedded Coder でのコード生成

モデルメニューの「コード」→「C/C++コード」→「コード生成オプション」よりコンフィギュレーションパラメータを起動し、コード生成を行います。

以下の設定を行い、コンフィギュレーションパラメータ — コード生成の「コード生成」を選択してください。

- ・コンフィギュレーションパラメータ — コード生成 — システムターゲットファイル「ert.tlc(Embedded Coder)」
- ・コンフィギュレーションパラメータ — コード生成 — 「コード生成のみ」にチェック
- ・コンフィギュレーションパラメータ — ソルバー — ソルバーオプション — タイプ「固定ステップ」

※ 使用する S-Function によって設定内容は異なります。コード生成時にエラーが発生する場合は、添付資料の設定例をご参考に、設定内容をご確認ください。



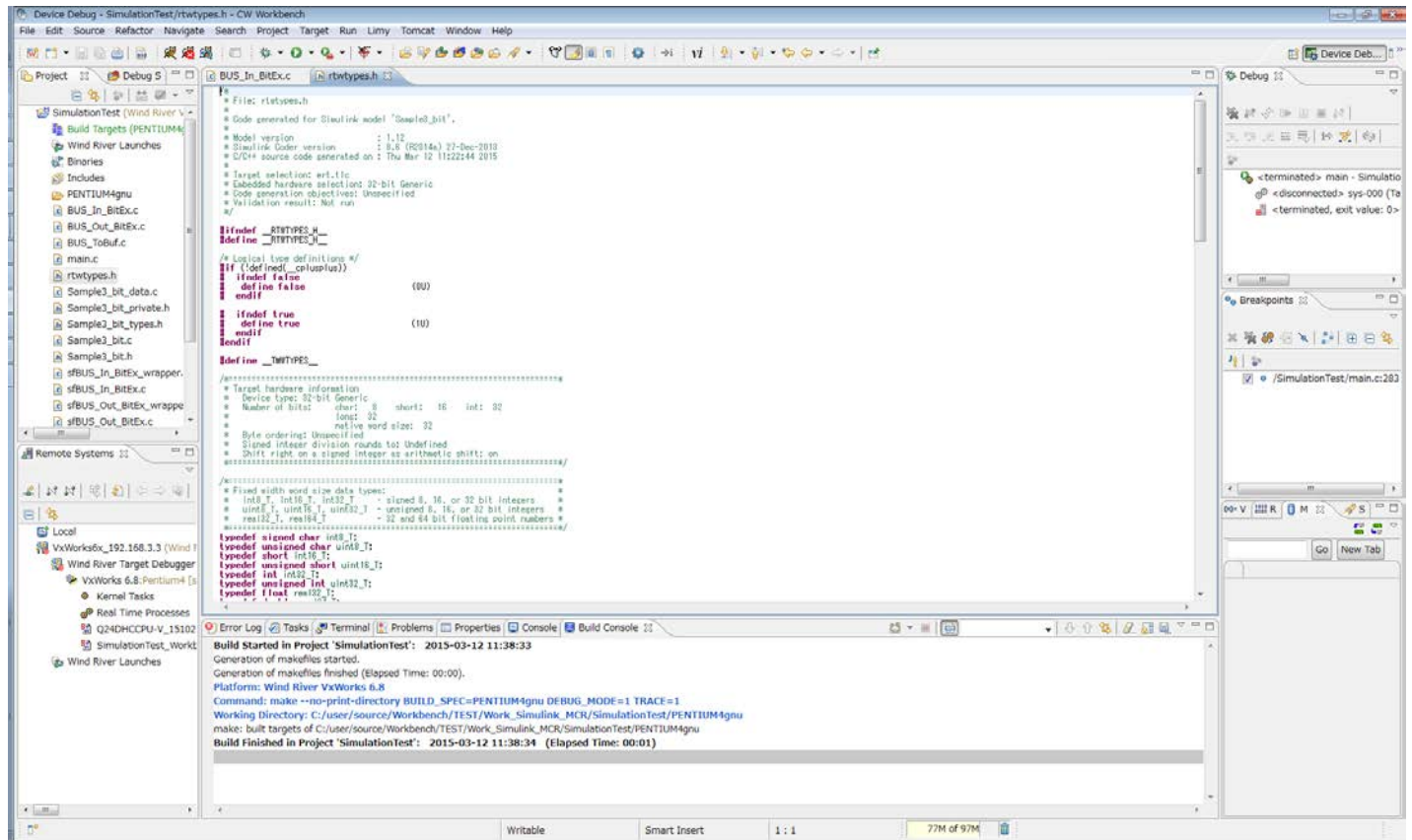
コード生成された C 言語ソース・インクルードファイルを CW Workbench でコンパイルし、C 言語コントローラに配置することで、C 言語コントローラ上で実行することができます。

ただし、MATLAB の Embedded Coder で出力されたメイン関数 (ert\_main.c) はサンプルプログラムですので、自作でメイン関数を作成してください。

なお、自動生成したプログラムは、MATLAB のインクルードを参照しているので、CW Workbench でコンパイルする場合は、関連するインクルードをリンク追加した上でコンパイル、ビルドしてください。詳細は 2.5 (2) をご参照ください。

## 2.5 CW Workbench でのビルド

C 言語コントローラ実行モジュールは、まず CW Workbench 上でビルドを行います。



CW Workbench 画面

CW Workbench にてプロジェクトを作成し、以下 (1)～(4) を実施してください。

(1) Simulink/Embedded Coder で自動生成したソースファイルの格納

MATLABで開発を行ったディレクトリ（\*\_ert\_rtwディレクトリ含む）のファイルのうち、以下のビルド対象外ファイルを除いたC言語ソース、インクルードファイルをCW Workbenchにてビルドしてください。

なお、ファイルは全てプロジェクトの同階層に格納してください。

### [ビルド対象外ファイル]

- ・ サンプルメイン関数 : ert\_main.c
- ・ C言語コントローラダミーインターフェースファイル : MdfFunc.h, QbfFunc.h, LibFunc.c
- ・ 拡張子c, h以外のファイル (\*.mexw32, \*.lib など)

(2) メイン関数の追加

MATLAB の Embedded Coder は、サンプルの mMain 関数(ert\_main.c)しか出力されませんので、関数：“モデル名”\_step を定周期で実行するメイン関数をご用意します。

本 S-Function を使用するには、プログラム開始時に初期化処理を行う必要があるため、メイン関数内にて初期化関数(SFunc\_Init())と終了関数(SFunc\_Final())をコールしてください。



### (3) MATLAB で使用した参照先インクルードファイルパスの追加

MATLAB の Embedded Coder より出力された C 言語ソースは、MATLAB 独自のインクルードファイルを含む場合があります。その場合は、MATLAB 用のインクルードファイルを CW Workbench の Include paths に追加してください。

追加方法 : Project → Properties → Build Properties → タブ[Build Paths] → Include paths → [Add...]

#### [インクルードファイル例]

- ・-I” MATLAB インストールフォルダ” /Simulink/include
- ・-I” MATLAB インストールフォルダ” /extern/include
- ・-I” MATLAB インストールフォルダ” /rtw/c/src
- ・-I” MATLAB インストールフォルダ” /toolbox/rtw/rtwdemos/EmbeddedCoderOverView <次行に続く>  
/CodeMetricFiles /PCG\_Eval\_CodeMetrics\_1

※モデルで使用しているライブラリにより、参照インクルードが異なる場合があります。

### (4) ビルドオプションの追加

CW Workbench の Project → Properties → Build Properties → タブ[Build Tools] → Command に『-D “RT” 』(ハイフン デイター スペース ダブルコーテーション アルティ ダブルコーテーション)を追加します。

※本プロジェクトは、実時間実行のプログラムになりますので、“RT” オプションを追加します。

C 言語コントローラの使用方法は、C 言語コントローラ設定・モニタツールオペレーティングマニュアル(sh081076)をご参照ください。



## 3. S-Function プログラムについて

### 3.1 S-Function 仕様

#### 3.1.1 バスインターフェース S-Function

C 言語コントローラから入出力ユニット、インテリジェント機能ユニットに直接アクセスする場合は、本 S-Function ライブラリを使用してください。

(1) ビットデバイス (X/Y) の読出し : sfBUS\_In\_BitEx

端子	端子名	行	データ型	説明
入力端子	sDevType	1	Int16	デバイスタイプ (X:1 / Y:2)
	sDevNo	1	Int16	入力デバイス先頭番号
	sSize	1	Int16	読込む Bit 数
出力端子	usData	可変※	uint16	読込んだ値 (0:OFF/1:ON)
	sRet	1	Int16	リターンコード (0:正常)
出力 C コード	sRet[0] = BUS_In_BitEx ( sDevType[0], sDevNo[0], sSize[0], usData );			
動作内容／ 注意事項	ビットデバイスを読み出します。 対象デバイスは X・Y デバイスです。			

(2) ビットデバイス (Y のみ) への書込み : sfBUS\_Out\_BitEx

端子	端子名	行	データ型	説明
入力端子	sDevNo	1	Int16	入力デバイス先頭デバイス番号
	sSize	1	Int16	読込む Bit 数
	usData	可変※	uint16	各デバイスに出力する値 (0:OFF/1:ON)
出力端子	sRet	1	Int16	リターンコード (0:正常)
出力 C コード	sRet[0] = BUS_Out_BitEx (sDevNo[0], sSize[0], usData);			
動作内容／ 注意事項	ビットデバイスへの書き込みを行います。 本 Function の対象ビットデバイスは Y デバイスのみです。他のビットデバイス使用時は sfBUS_WriteDevice をご使用ください。			

(3) 内部デバイスデータの読出し : sfBUS\_ReadDevice

端子	端子名	行	データ型	説明
入力端子	sDevType	1	Int16	内部デバイスタイプ (X:1 / Y:2)
	sDevNo	1	Int16	入力デバイス先頭番号
	sSize	1	Int16	読込む Bit 数
出力端子	usData	可変※	uint16	読込んだ値 (0:OFF/1:ON)
	sRet	1	Int16	リターンコード (0:正常)
出力 C コード	sRet[0] = BUS_ReadDevice (sDevType[0], ulDevNo[0], ulSize[0], usDevData);			
動作内容／ 注意事項	内部デバイスの読み込みを行います。 X, Y デバイスの読み込みを行いたい場合は sfBUS_In_BitEx をご使用ください。			

(4) 内部デバイスデータへの書込み : sfBUS\_WriteDevice

端子	端子名	行	データ型	説明
入力端子	sDevType	1	Int16	内部デバイスタイプ ( )
	sDevNo	1	Int16	入力デバイス先頭デバイス番号
	sSize	1	Int16	読込む Bit 数
	usData	可変※	uint16	各デバイスに出力する値 (0:OFF/1:ON)
出力端子	sRet	1	Int16	リターンコード (0:正常)
出力 C コード	sRet[0] = BUS_WriteDevice (sDevType[0], ulDevNo[0], ulSize[0], usData);			
動作内容／ 注意事項	内部デバイスへの書き込みを行います。 Y デバイスへ書き込みを行いたい場合は sfBUS_Out_BitEx をご使用ください。			



## (5) バッファメモリの読出し : sfBUS\_FromBuf

端子	端子名	行	データ型	説 明
入力端子	usIoNo	1	Int16	先頭 I/O 番号 ÷ 16 先頭 I/O 番号は PC パラメータ I/O 割付設定の先頭 XY の値を指す。例えば先頭 XY が 0010 の場合は、usIoNo=1
	ulOffset	1	Int32	バッファメモリのアドレス
	ulSize	1	Int32	読込むワード数
出力端子	usData	可変※	uint16	読込んだ値 (WORD 型の配列)
	sRet	1	Int16	リターンコード (0:正常)
出力 C コード	sRet[0] = BUS_FromBuf (usIoNo[0], ulOffset[0], sSize[0], usData);			
動作内容／ 注意事項	インテリジェント機能ユニットのバッファメモリから読み出しを行います。			

## (6) バッファメモリへの出力 : sfBUS\_ToBuf

端子	端子名	行	データ型	説 明
入力端子	usIoNo	1	Int16	先頭 I/O 番号 ÷ 16
	ulOffset	1	Int32	バッファメモリのアドレス
	ulSize	1	Int32	書込むワード数
	usData	可変※	uint16	各デバイスに出力する値
出力端子	sRet	1	Int16	リターンコード (0:正常)
出力 C コード	sRet[0] = BUS_ToBuf (usIoNo[0], ulOffset[0], ulSize[0], usData);			
動作内容／ 注意事項	インテリジェント機能ユニットのバッファメモリへの書き込みを行います。 書き込みアドレスは、各ユニットの仕様をよくご確認の上指定してください。			

※ 行可変の端子は、S-Function Builder で指定した行数（配列数）のデータを取得するため、  
「sSize<S-Function Builder の usData 行数」となるよう設定が必要です。  
各出力端子のリターンコード sRet の値は、バスインターフェース関数ヘルプのエラー一覧を参照願います。

## 3.2 プログラムファイル概要

S-Function に使用するプログラムの概要は以下となります。

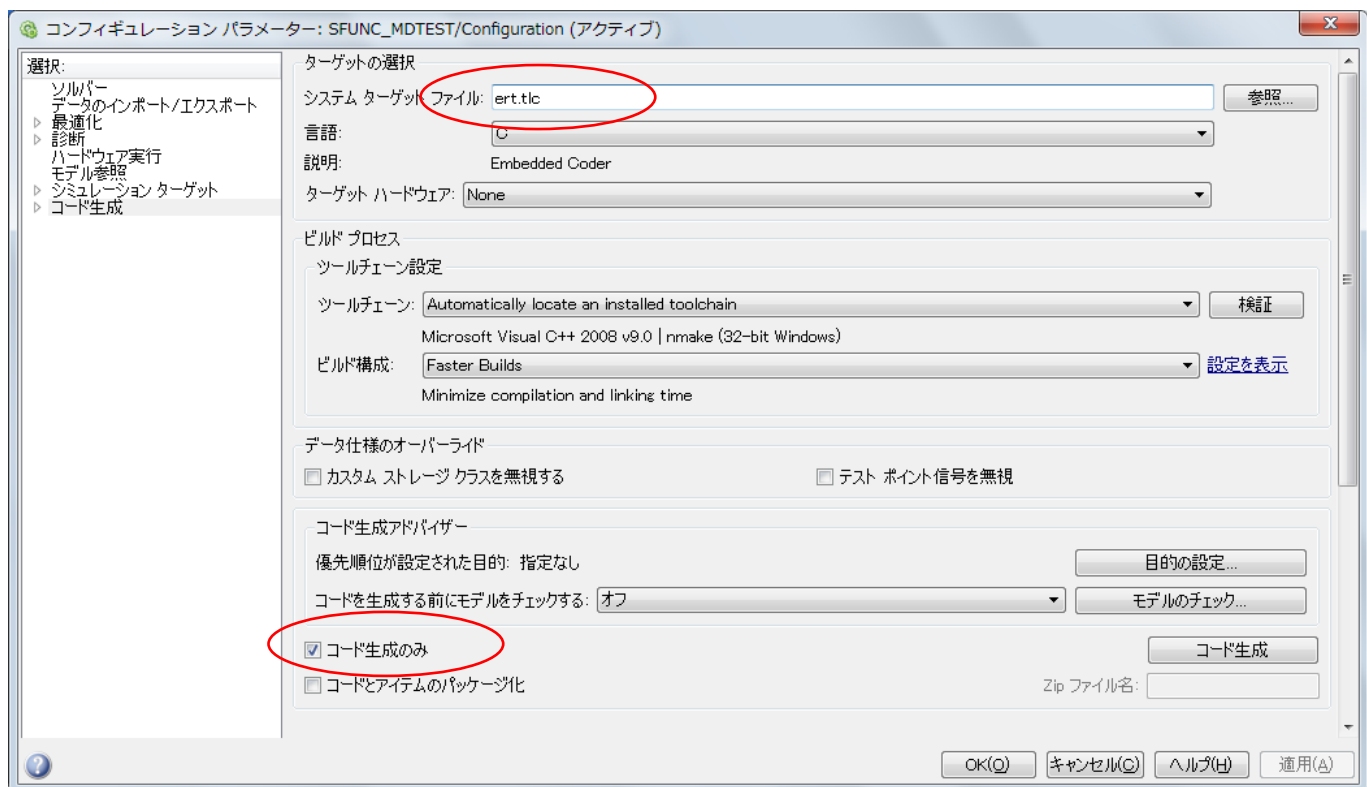
ファイル名	説 明	備 考
BUS_FromBuf.c	sfBUS_FromBuf の処理プログラムです。	
BUS_In_BitEx.c	sfBUS_In_BitEx の処理プログラムです。	
BUS_Out_BitEx.c	sfBUS_Out_BitEx の処理プログラムです。	
BUS_ReadDevice.c	sfBUS_ReadDevice の処理プログラムです。	
BUS_ToBuf.c	sfBUS_ToBuf の処理プログラムです。	
BUS_WriteDevice.c	sfBUS_WriteDevice の処理プログラムです。	
LibFunc.c	Simulink のデバッグ実行時に使用する、C 言語コントローラインターフェースのダミープログラムファイルです。	CW Workbench では、このファイルは使用しないでください
QbfFunc.h	ダミープログラム (QBF 関数) のヘッダファイルです。	CW Workbench では、このファイルは使用しないでください
SFuncCom.h	S-Function の処理プログラム用ヘッダファイルです。	
SFuncLib.c	共通関数を記載したファイルです。	記載関数：初期化関数、終了関数



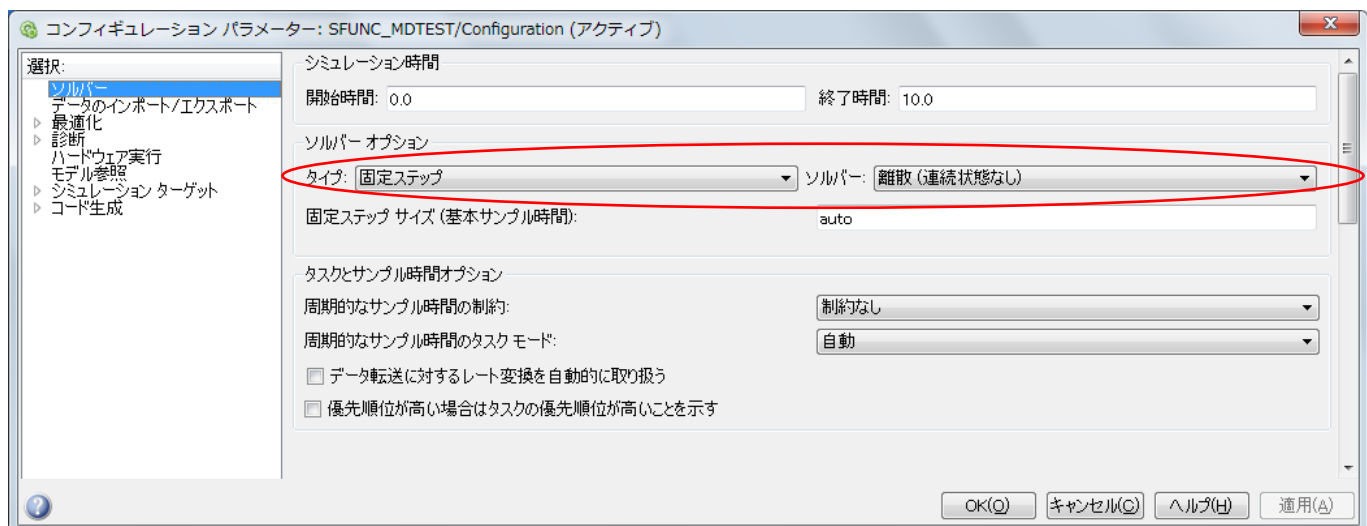
## 添付資料 コンフィギュレーションパラメータ設定例

プログラムコード生成時にエラーが発生した場合は、以下設定をご確認ください。

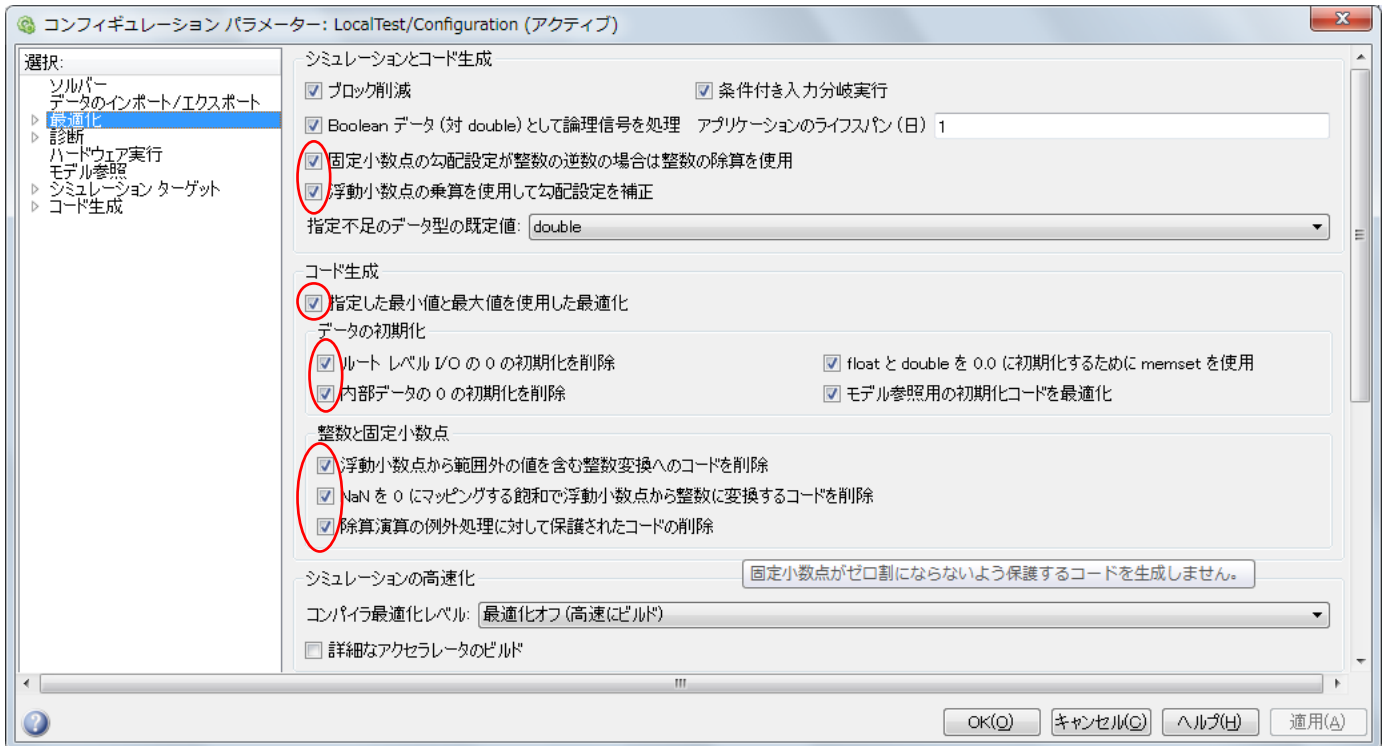
[コード生成] 最初に設定してください



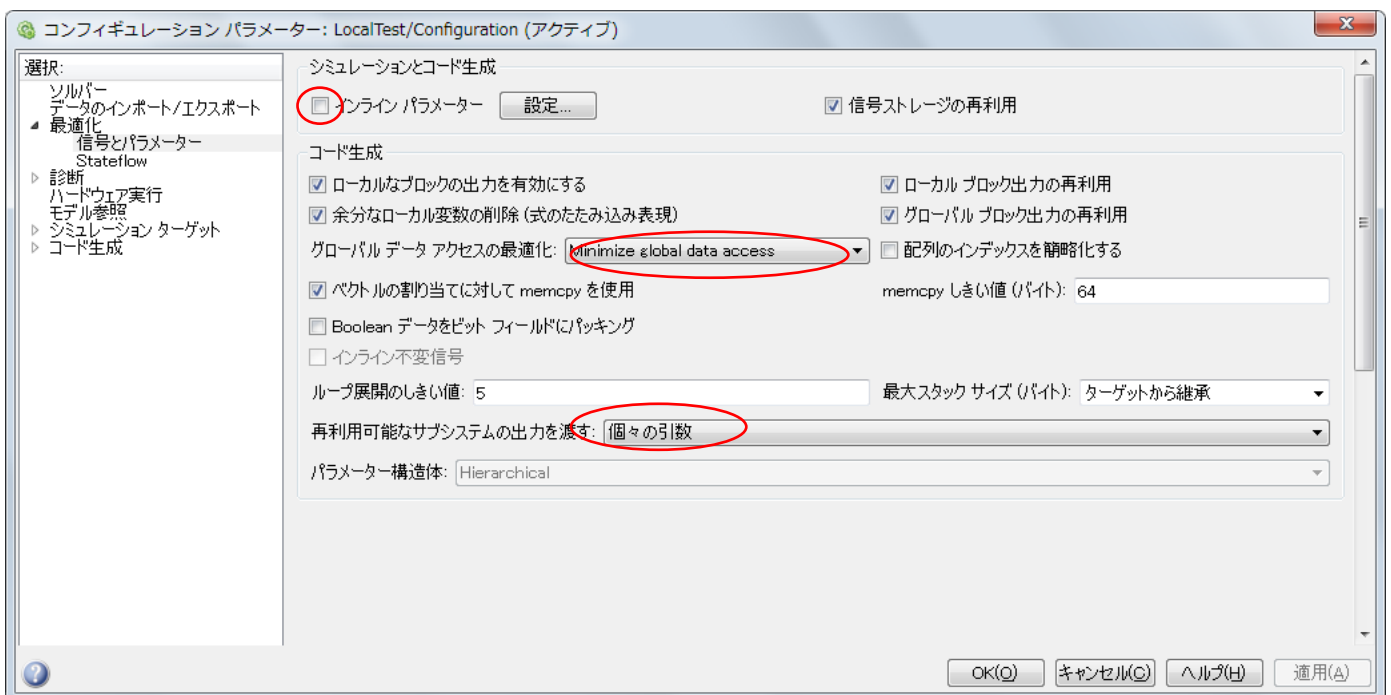
[ソルバー]



## [最適化]

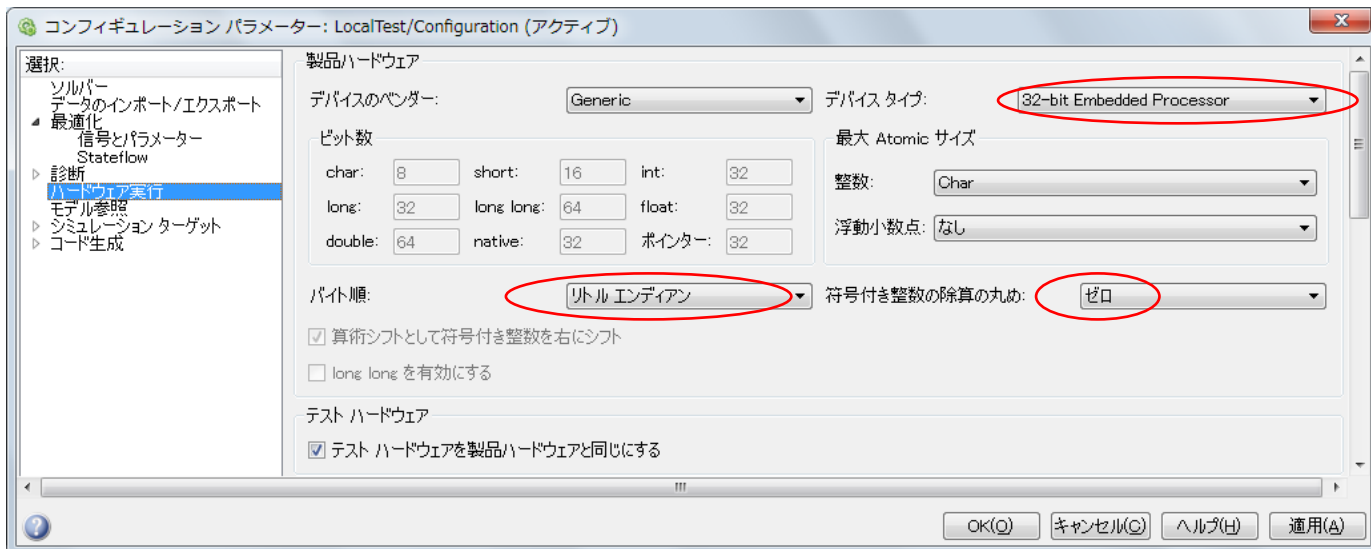


## [最適化] → [信号とパラメータ]

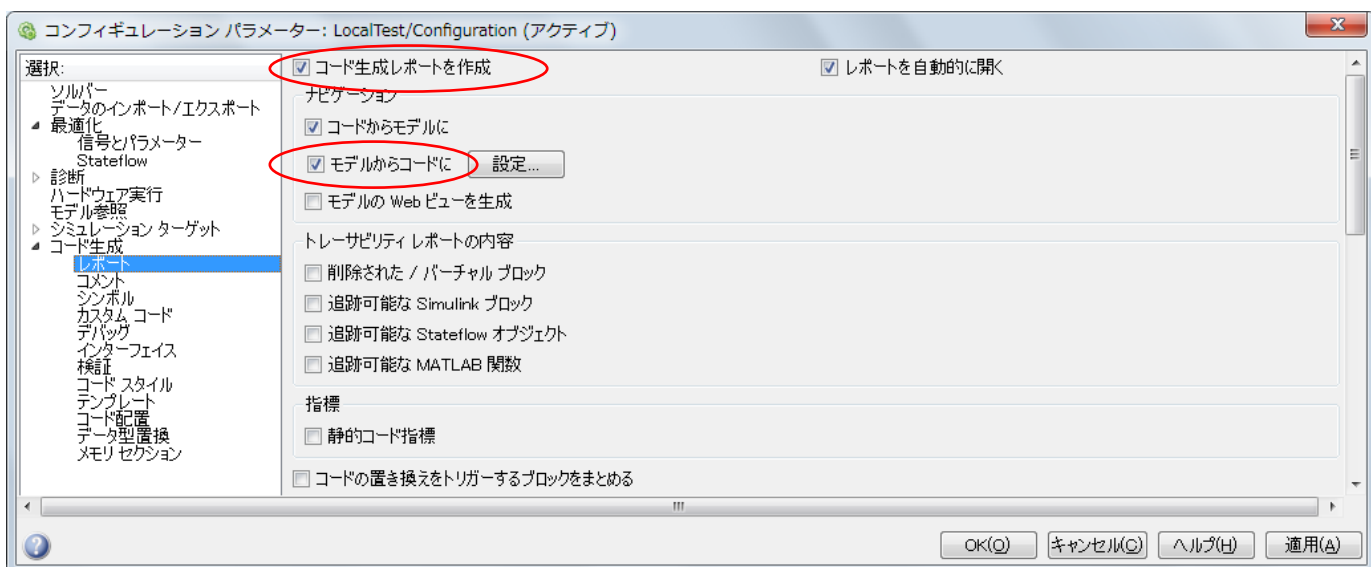




## [ハードウェア実行]



## [コード生成]→[レポート]



[コード生成]→[シンボル]

コンフィギュレーション パラメーター: SFUNC\_MDTEST/Configuration (アクティブ)

選択:

- ソルバー
- データのインポート/エクスポート
- 最適化
  - 信号とパラメーター
  - Stateflow
- 診断
  - ハードウェア実行
  - モデル参照
- シミュレーション ターゲット
  - コード生成
    - レポート
    - コメント
    - シンボル
    - カスタム コード
    - デバッグ
    - インターフェイス
    - 検証
    - コード スタイル
    - テンプレート
    - コード配置
    - データ型置換
    - メモリセクション

自動生成される識別子の命名規則

識別子の書式の制御

グローバル変数: \$R\$N\$M

グローバルなタイプ: \$N\$R\$M

グローバルなタイプのフィールド名: \$N\$M

サブシステムのメソッド: \$R\$N\$M\$F

サブシステムのメソッド引数: rt\$N\$M

ローカルな一時変数: \$N\$M

ローカルなブロックの出力変数: rtb\_\$N\$M

定数のマクロ: \$R\$N\$M

共有ユーティリティ: \$N\$C

マングルの最小の長さ: 1

識別子の最大の長さ: 31

システム生成の識別子: 短縮形

スカラーのインライン パラメーターの生成: リテラル

Simulink データ オブジェクトの命名規則

信号名: なし

パラメーター名: なし

#define 定義名: なし

OK(Q) キャンセル(Q) ヘルプ(H) 適用(A)

[コード生成]→[インターフェイス]

コンフィギュレーション パラメーター: LocalTest/Configuration (アクティブ)

選択:

- ソルバー
- データのインポート/エクスポート
- 最適化
  - 信号とパラメーター
  - Stateflow
- 診断
  - ハードウェア実行
  - モデル参照
- シミュレーション ターゲット
  - コード生成
    - レポート
    - コメント
    - シンボル
    - カスタム コード
    - デバッグ
    - インターフェイス
    - 検証
    - コード スタイル
    - テンプレート
    - コード配置
    - データ型置換
    - メモリセクション

ソフトウェア環境

標準の数学ライブラリ: C89/C90 (ANSI)

コード置換ライブラリ: None

共有コードの配置: 自動

サポート:
 

- ☒ 浮動小数点数
- ☐ 非有限数
- ☐ 複素数
- ☒ 絶対時間
- ☐ 連続時間
- ☐ インラインでない S-Function
- ☐ 可変サイズの信号

マルチワード タイプの定義: システム定義

コード インターフェイス

コード インターフェイスのパッケージ化: Nonreusable function

☐ クラシック コール インターフェイス

☒ 1 つの出力/更新関数

☒ 終了関数の生成

グリブロッサの条件を生成: ローカル設定を使用

☒ リアルタイム モデルのデータ構造内のエラー ステータスを非表示にする

☐ 信号と状態の構造の統合

モデルの開数を設定

データ交換

☐ MAT ファイルのログ

インターフェイス: MATLAB の .mat ファイルへのデータのログを取るためのコードを生成します。

OK(Q) キャンセル(Q) ヘルプ(H) 適用(A)

※MATLAB 及び Simulink は、The MathWorks, Inc. の登録商標です。