SIMULINK

Dynamic System Simulation for MATLAB®

Modeling

Simulation

Implementation





Version 4

サイバネットシステム株式会社との連絡法



Using Simulink

© COPYRIGHT 1990 - 2000 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by or for the federal government of the United States. By accepting delivery of the Program, the government hereby agrees that this software qualifies as "commercial" computer software within the meaning of FAR Part 12.212, DFARS Part 227.7202-1, DFARS Part 227.7202-3, DFARS Part 252.227-7013, and DFARS Part 252.227-7014. The terms and conditions of The MathWorks, Inc. Software License Agreement shall pertain to the government's use and disclosure of the Program and Documentation, and shall supersede any conflicting contractual terms or conditions. If this license fails to meet the government's minimum needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to MathWorks.

MATLAB, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop are registered trademarks, and Target Language Compiler is a trademark of The MathWorks, Inc.

Other product or brand names are trademarks or registered trademarks of their respective holders.

Printing History:	November 1990	First printing
	December 1996	Revised for Simulink 2
	January 1999	Revised for Simulink 3 (Release 11)
	November 2000	Revised for Simulink 4 (Release 12)
	January 2001	翻訳

本書の内容の一部あるいは全部を無断で転載、複製、複写することを禁じます。 本書の内容は予告なく変更することがあります。

目次

Getting Started

読者へ Simulink とは マニュアルの使用法	 -2 -2 -3
関連プロダクト	 -6

クイックスタート

2

1

デモモデルの実行	2-2
デモの説明	2-3
試してみるべきこと	2-4
このデモが示すもの	2-4
その他の有効なデモ	2-5
簡単なモデルの作成 2	2-6
Simulink の設定	-15
Simulink Preferences 2-	·15

Simulink の機能

3

Simulink とは	3-2
ダイナミックシステムのモデリング ブロック線図	3-3 3-3
ブロック	3-3

状態 :	3-4
システム関数	3-4
ブロックパラメータ	3-5
連続ブロックと離散ブロック :	3-6
サブシステム	3-6
カスタムブロック	3-6
信号:	3-7
データタイプ	3-7
ソルバ	3-8

ダイナミックシステムのシミュレーション 3-9

モデルの初期化フェーズ	3-9
モデル実行フェーズ	. 3-10
時間ステップ毎の処理	. 3-10
ブロックのアップデート順の決定	. 3-11
Atomic サブシステムと仮想サブシステム	. 3-13
ソルバ	. 3-13
ゼロクロッシングの検出	. 3-14
代数ループ	. 3-18

離散システムのモデル化とシミュレーション	3-23
離散ブロック	3-23
サンプル時間	3-23
純粋な離散システム	3-23
マルチレートシステム	3-24
離散システムのステップサイズの決定	3-24
サンプル時間の伝播	3-26
不变定数	3-27
連続および離散混在システム	3-28

モデルの作成

$4 \lceil$

Simulink の起動	4-2
新規モデルの作成	4-3
既存のモデルの編集	4-3
Simulink コマンドの実行	4-4

Simulink ウィンドウ	4-5
オプジェクトの選択	4-8
1つのオブジェクトの選択	4-8
複数のオブジェクトの選択	4.8
	4-0
プロック	4-10
ブロックのデータ情報	4-10
バーチャルブロック	4-10
ウィンドウ間でのブロックのコピーと移動	4-11
モデル内でのブロックの移動	4-13
モデル内でのブロックの複製	4-13
ブロックパラメータの指定	4-14
ブロック固有のパラメータの設定	4-14
ブロックプロパティダイアログボックス	4-15
ブロックの削除	4-17
ブロックの方向の変更	4-17
ブロックのサイズ変更	4-18
ブロック名の操作	4-18
ブロックアイコンの下側にパラメータを表示	4-19
ブロックの切り離し	4-20
ブロックプライオリティの割り当て	4-20
ブロック実行順序の表示	4-21
ドロップシャドウの使用法	4-21
サンプル時間の色分け	4-22
プロックの接続	4-24
ブロック間でのラインの描画	4-24
分岐線の描画	4-25
ラインセグメントの描画	4-25
ラインセグメントの移動	4-26
ラインのセグメントへの分割	4-27
ラインの頂点の移動	4-28
ラインにブロックを挿入	4-28
信号の利用	4-30
信号について	4-30
バス信号	4-32
信号に関する用語集	4-33
出力信号の大きさの決定	4-34

信号とパラメータの大きさの規則	4-35
入力とパラメータのスカラ拡張	4-36
複素数値信号の利用	4-37
信号の接続のチェック	4-38
信号の表示オプションの設定	4-38
信号の名前	4-39
信号ラベル	4.39
バーチャル信号で表わされる信号の表示	4.40
信号プロパティの設定	4-40
信号プロパティダイマログボックフ	4-40
	4-41
计图	
注朳	4-44
テーダダイブの機能	4-45
	4-45
フロックがサホートするテータと数の信号タイフ	4-46
ブロックパラメータのデータタイプの指定	4-46
特定のデータタイプの信号の作成	4-47
端子のデータタイプの表示	4-47
データタイプの伝達	4-47
データタイプの法則	4-48
厳密な Boolean 型の検査を有効にする	4-49
信号の型変換	4-49
パラメータの型変換	4-49
データオブジェクトの機能	4-51
データオブジェクトクラス	4-51
データオブジェクトの作成	4-52
データオブジェクトプロパティへのアクセス	4-53
データオブジェクトメソッドの呼び出し	4-53
データオブジェクトの保存とロード	4-54
Simulink モデル内のデータオブジェクトの使用	4-54
データオブジェクトクラスの作成	4-56
Simulink データエクスプローラ	4-61
マウスとキーボードの動作のまとめ	4-63
サプシステムの作成	4-66
Subsystem ブロックの追加によるサブシステムの作成	4-66
	- 50

既存のブロックのグループ化によるサブシステムの作成	4-67
モデルナビゲーションコマンド	4-68
ウィンドウの再利用	4-68
サブシステム端子のラベリング	4-69
サブシステムへのアクセス制御	4-70
コールバックルーチンの使用法	4-71
コールバックのトレーシング	4-71
モデルコールバックパラメータ	4-71
ブロックコールバックパラメータ	4-72
モデル作成のヒント	4-76
ライプラリ	4-77
專門用語	4-77
ライブラリの作成	4-77
ライブラリの修正	4-78
ライブラリリンクの作成	4-78
ライブラリリンクを無効にする	4-79
リンクしているサブシステムの変更	4-79
リンクサブシステムの変更をライブラリに反映させる	4-80
リンクされたブロックのアップデート	4-80
ライブラリブロックに対するリンクの解除	4-80
参照ブロックに対するライブラリブロックの検出	4-81
ライブラリのリンク状態	4-81
ライブラリリンクの表示	4-82
ライブラリブロックについての情報の取得	4-82
ブロックライブラリの参照	4-83
ライブラリブラウザにライブラリを追加	4-85
方程式のモデル化	4-87
掲氏から華氏への変換	4-87
簡単な連続システムのモデル化	4-88
モデルの保存	4-90
ブロック線図の印刷	4-91
Print ダイアログボックス	4-91
Print コマンド	4-92

用紙の大きさと方向の指定ダイアグラムの位置と大きさ	4-93 4-94
モデルの検索とブラウズ	4-95
オブジェクトの検索	4-95
モデルブラウザ	4-100
モデルのバージョン管理	4-106
カレントユーザの指定	4-106
モデルプロパティダイアログ	4-109
モデル更新履歴の作成	4-113
ハーション官埋フロハティ	4-114 4-116

シミュレーションの実行

はじめに	5-2
メニューコマンドの使用法	5-2
コマンドラインからのシミュレーションの実行	5-3
メニューコマンドを用いたシミュレーションの実行	5-4
シミュレーションパラメータの設定とソルバの選択	5-4
シミュレーションパラメータの適用	5-4
シミュレーションの開始	5-4
Simulation Diagnostics ダイアログボックス	5-6
Simulation Parameters ダイアログボックス	5-8
Solver $^{\sim}$	5-8
Workspace I/O $^{\sim}$ - $^{\circ}$	-18
Diagnostics $^{-y}$.	-25
アドバンスドパネル	-29
シミュレーションの性能と精度の改良 5-	
	-34
シミュレーションの高速化 5-	34 34

5

コマンドラインからのシミュレーションの実行	5-36
sim コマンドの利用	5-36
set_param コマンドの利用	5-36
sim	5-38
simplot	5-41
simset	5-43
simget	5-47

シミュレーション結果の解析

6

7

出力軌跡の表示 6-2 Scope ブロックの使用法 6-2 出力変数の使用法 6-2 To Workspace ブロックの使用法 6-2	2 2 2 3
線形化	4
平衡点の決定	7 9
trim	2

マスクを使ったプロックのカスタマイズ

はじめに	7-2
マスクされたサンプルサプシステム マスクダイアログボックスのプロンプトの作成	7-3 7-4 7-6 7-6
Mask Editor: 概要	7-8

Initialization $^{\mathbf{N}}-\vec{\mathcal{Y}}$	7-9
プロンプトと関連する変数	7-9
コントロールのタイプ	7-11
マスクされたブロックパラメータに対するデフォルト値	7-13
変更可能なパラメータ	7-13
初期化コマンド	7-14
Icon ページ	7-17
ブロックアイコン上でのテキストの表示	7-17
ブロックアイコン上でのグラフィックスの表示	7-19
マスク上でのイメージの表示	7-20
ブロックアイコン上での伝達関数の表示	7-21
アイコンプロパティの制御	7-22
Documentation $^{\sim}-^{:}$	7-26
Mask Type フィールド	7-26
Block Description $7 - \mu F$	7-26
Mask Help Text フィールド	7-27
Self-Modifying マスクブロックの作成	7-28
マスクブロックに対する動的ダイアログの作成	7-29
マスクブロックダイアログのパラメータの設定	7-29
定義済みマスクダイアログパラメータ	7-30

条件付きで実行されるサプシステム

はじめに	8-2
Enabled サプシステム	8-3
Enabled サブシステムの作成	8-3
Enabled サブシステムに組み込み可能なブロック	8-5
Triggered サプシステム	8-8
Triggered サブシステムの作成	8-9
Function-Call サブシステム 8	3-10

8

Triggered サブシステムに組み込み可能な ブロック 8-:

Triggered and Enabled サブシステム	8-12
Triggered and Enabled サブシステムの作成	8-12
Triggered and Enabled サブシステムのサンプル	8-13
交互に実行するサブシステムの作成	8-13

9

プロックリファレンス

各フロックのリファレンスページの内容	9-2
	9-3
Abs	9-11
Algebraic Constraint	9-13
Backlash	9-15
Band-Limited White Noise	9-19
Bitwise Logical Operator	9-21
Bus Selector	9-25
Chirp Signal	9-27
Clock	9-29
Combinatorial Logic	9-31
Complex to Magnitude-Angle	9-34
Complex to Real-Imag	9-35
Configurable Subsystem	9-36
Constant	9-40
Coulomb and Viscous Friction	9-42
Data Store Memory	9-44
Data Store Read	9-46
Data Store Write	9-47
Data Type Conversion	9-49
Dead Zone	9-51
 Demux	9-53
Derivative	9-58
Digital Clock	9-60
Direct Look-Un Table (n-D)	9-61
Discrete Filter	9-67

Discrete Pulse Generator
Discrete State-Space
Discrete-Time Integrator
Discrete Transfer Fcn
Discrete Zero-Pole
Display
Dot Product
Enable
Fcn
First-Order Hold
From
From File
From Workspace
Function-Call Generator 9-106
Gain
Goto
Goto Tag Visibility
Ground
Hit Crossing
IC
Inport
Integrator
Logical Operator
Look-Up Table
Look-Up Table (2-D) 9-135
Look-Up Table (n-D)
Magnitude-Angle to Complex 9-143
Manual Switch
Math Function 9-146
MATLAB Fcn
Matrix Gain
Memory
Merge
MinMax
Model Info
Multiport Switch
Mux
Outport
Product
Probe 9-173

Pulse Generator	·175
Quantizer	·177
Ramp 9-	·179
Random Number	·181
Rate Limiter	·183
Real-Imag to Complex	·185
Relational Operator	·187
Relay	·189
Repeating Sequence	·191
Reshape	·193
Rounding Function	·195
Saturation	·196
Scope	·198
Selector	-208
S-Function	-212
Sign 9.	-214
Signal Generator	-215
Sine Wave	-218
Slider Gain	-221
State-Space	-223
Step	-225
Stop Simulation	-227
Subsystem	-228
Sum	-232
Switch	-235
Terminator	-237
To File	-238
To Workspace	-240
Transfer Fcn	-243
Transport Delay	-246
Trigger	-248
Trigonometric Function	-250
Uniform Random Number	-252
Unit Delay	-254
Variable Transport Delay	-256
Width 9-	-259
XY Graph	-260
Zero-Order Hold	-262
Zero-Pole	-263

10 [

HIX.	10.2
	10-2
コマンドにパラメータを指定する方法	10-3
Simulink オブジェクトにパスを指定する方法	10-3
add_block	10-4
add line	10-5
bdclose	10-6
bdroot	10.7
alosa system	10-7
	10-0
delete_block	10-10
delete_line	10-11
find_system	10-12
gcb	10-17
gcbh	10-18
gcs	10-19
get naram	10-20
pow system	10 20
new_system	10-22
open_system	10-23
replace_block	10-24
save_system	10-26
set_param	10-27
simulink	10-29

Simulink デバッガ

11 [

デバッガの起動	11-3
シミュレーションの開始	11-4
デバッガのコマンドラインインタフェースを使って ブロックインデックスについて MATLAB ワークスペースへのアクセス	11-6 11-6 11-6
オンラインヘルプ	11-7

シミュレーションの実行	. 11-8
シミュレーションの継続	. 11-8
シミュレーションをノンストップで実行	. 11-9
	11.9
	. 11-7
时间人ナツノ母に夫1]	11-10
プレークポイントの設定	11-11
ブロックでブレークポイントを設定	11-12
時間ステップでブレークポイントを設定	11-14
北方限値で停止	11_14
非日限値でげエ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	11-14
ステップサイスを制限しにステップで停止	11-14
セロクロッシングにおいて停止	11-14
シミュレーションに関する情報の表示	11-16
ブロック 1/0 の表示	11-16
2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	11-10
	11-10
	11-18
積分情報の表示	11-19
モデルに関する情報の表示	11-20
モデルのブロック実行順序の表示	11_20
	11-20
クロックの夜小	11-20
デバッガコマンドリファレンス	11-24
ashow	11-27
atrace	11-28
hafter	11.29
handr	11-27
oreak	11-50
bshow	11-31
clear	11-32
continue	11-33
disp	11-34
heln	11-35
ichow	11 26
ISIIOW	11-50
minor	11-37
nanbreak	11-38
next	11-39
probe	11-40
- quit	11-41
1	11.42

slist 11-4	3
states 11-4	4
systems 11-4	5
status 11-4	6
step 11-4	7
stop 11-4	8
tbreak 11-4	9
trace	60
undisp 11-5	1
untrace	2
xbreak 11-5	3
zcbreak 11-5	4
zclist	5

Performance Tools

12 [

Simulink Performance Tools オプション	12-2
Simulink Accelerator	12-3
機能方法	12-3
Simulink Accelerator の実行方法	12-4
モデル構造の変更の取り扱い	12-5
Accelerator モードの性能向上	12-6
速度の改良を示さないブロック	12-7
Simulink Accelerator を Simulink Debugger で利用	12-8
Simulink Accelerator とのプログラムインタフェース	12-9
性能の比較	12-10
Simulink Accelerator Build プロセスのカスタマイズ	12-10
Simulink Accelerator での S- ファンクションの実行の制御	12-11
Model Differencing Tool	12-13
表示オプション	12-15
モデルの差分をレポート	12-15
プロファイラ	12-17
どのようにしてプロファイラが働くか	12-17

プロファイラを利用可能にする	12-19
シミュレーションプロファイル	12-20
Model Coverage Tool(モデル補償範囲ツール)	12-23
どのようにして Model Coverage Tool が働くか	12-23
Model Coverage Tool の使用	12-23
テストケースの作成と実行	12-24
カバレージレポート	12-26
カバレージの設定ダイアログボックス	12-29
Model Coverage コマンド	12-31

モデルとブロックパラメータ

はじめに
モデルパラメータ A-3
共通プロックのパラメータ A-7
プロック固有のパラメータ A-10
マスクパラメータ A-20

モデルファイルフォーマット

$B \lceil$

A

モデルファイルの内容	B-2
Model セクション	B-3
BlockDefaults セクション	B-3
AnnotationDefaults セクション	B-3
System セクション	B-3

1

Getting Started

読者へ Simulink とは マニュアルの使用法											1-2 1-2 1-3
関連プロダクト											1-6



Simulink[®] へようこそ! この数年の間に Simulink は、ダイナミックシステムのモ デル化とシミュレーション用として学界、産業界でもっとも広く普及したソフト ウェアパッケージとなりました。

Simulink により試験作業が容易になります。何もないところからモデルを作成し たり、既存のモデルを取り出して、それに追加したりすることが容易に行えま す。シミュレーションは対話型で行われるので、実行中にパラメータを変更した ときの変化を即座に見ることができます。MATLAB®のすべての解析ツールに瞬 時にアクセスできるので、結果を取り出してそれらを分析し、視覚化することが できます。問題を提起してモデル化し、どうなるかを観察することが容易に行え る環境を通じて、モデル化とシミュレーションのおもしろさを体得してくださ い。

Simulink を用いて、理想化した線形モデルの範囲を越えて、摩擦、空気抵抗、ギ ヤのすべり量、ハードストップ、実世界の種々の現象を記述するその他の事柄を 計算に取り入れて、より現実的な非線形モデルを探求することができます。 Simulink は、コンピュータを、自動車のクラッチシステム、航空機の翼の揺れ、 捕食者 - 被食者モデルのダイナミクス、経済における通貨供給の効果、他の方法 では不可能または非現実的であるようなシステムのモデル化と解析のための実験 室に変えます。

Simulink は、また実際的でもあります。世界中で何千というエンジニアが Simulink を用いて実際の問題のモデル化や解決を行っており、このツールの知識 は、専門分野の仕事をしている限り、常に十分な貢献をしてくれます。

このソフトウェアの探索を是非楽しんでください。

Simulink とは

Simulink は、ダイナミックシステムのモデル化、シミュレーション、解析を行う ためのソフトウェアパッケージです。Simulink は、連続時間、離散時間または連 続時間と離散時間のハイブリッドでモデル化した線形システムおよび非線形シス テムをサポートします。システムはマルチレート、すなわち異なるレートでサン プリングや更新が行われる種々のパーツから構成されていても構いません。

モデル化のために Simulink は、マウスのクリックアンドドラッグ操作を用いて、 ブロック線図としてモデルを作成するためのグラフィカルユーザインタフェース (GUI)を提供します。このインタフェースを使用すると、紙と鉛筆を用いて描く のと同じように(あるいは、ほとんどのテキストブックに描かれているように) モデルを描画することができます。これは、言語やプログラムで微分方程式や差 分方程式を作成する必要があったこれまでのシミュレーションパッケージとは大きく異なります。Simulink には、出力表示、入力源、線形および非線形成分、結合などの包括的ブロックライブラリが含まれています。また、独自のブロックをカスタマイズしたり、作成したりすることもできます。ユーザ定義ブロックの作成に関する詳細は、Writing S-Functions ガイドをご参照ください。

モデルは階層構造になっているので、トップダウン、およびボトムアップの両方 のアプローチでモデルを作成することができます。システムは、まず高いレベル で表示し、つぎにブロックをダブルクリックして下のレベルに移行して、モデル をより詳細に見ることができます。このアプローチによってモデルがどのように 構成され、その各パーツがどのように相互作用するかを洞察することができま す。

モデルの定義が終わると、積分手法を Simulink のメニューから選択するか、あ るいは MATLAB のコマンドウィンドウにコマンドを入力して、モデルのシミュ レーションを行うことができます。メニューは、対話型の作業に特に便利です が、複数のシミュレーションをバッチ型で実行する場合(たとえばモンテカルロ シミュレーションを行う場合や、ある範囲の値全体に対してパラメータを動かし たい場合)は、コマンドラインアプローチが非常に有効です。スコープや他の表 示ブロックを使うと、シミュレーションの実行中にシミュレーション結果を見る ことができます。さらに、パラメータを変更して、"what if(こうしたらどうだろ う)" 探求のために、即座にその結果を見ることができます。シミュレーション 結果は、後処理や視覚化のために MATLAB ワークスペースに転送することがで きます。

モデル解析ツールには、MATLAB コマンドラインからアクセスできる線形化 ツールと平衡点算出ツールが含まれており、さらに MATLAB およびアプリケー ションツールボックス内の多くのツールを併用できます。また、MATLAB と Simulink は統合化されているので、任意の時点のいずれの環境内でも、モデルの シミュレーションや解析、修正を行うことができます。

マニュアルの使用法

Simulink はグラフィカルで対話型なので、すぐそれを起動して試されることをお 薦めします。

Simulink をすみやかに使いはじめるために役立つ導入については、第2章の"デ モモデルの実行"を参照してください。モデルをながめて、興味のあるブロック をダブルクリックしてください。そうすることによって、Simulink がどのように 動作するかをすぐに感じ取ることができます。モデル作成について速習したい場 合には、第2章の"簡単なモデルの作成"を参照してください。 Simulinkの記述的な導入については、第3章の"Simulinkの機能"を参照してください。この章では、Simulinkモデルの作成方法と実行方法を理解するのに必要となる主な概念を紹介します。

第4章の"モデルの作成"では、モデルの作成と変更方法を詳しく説明します。モ デルの保存と印刷方法についても説明し、役立つヒントを提供します。

第5章の"シミュレーションの実行"では、Simulink がどのようにシミュレーションを実行するかについて説明します。シミュレーションパラメータとシミュレーションに使用する積分ソルバについて、問題に対する適切なソルバを選択するのに役立つ、各ソルバの長所と短所を含めて説明します。

第6章の"シミュレーション結果の解析"では、シミュレーション結果の表示と 分析に役立つ Simulink と MATLAB の機能について説明します。

第7章の"マスクを使ったブロックのカスタマイズ"では、独自のブロックを作成し、マスク機能を使用してその外観と使用方法をカスタマイズする方法について 説明します。

第8章の"条件付きで実行されるサブシステム"では、信号のトリガイベントに よって実行方法が異なるサブシステムについて説明します。

第9章の"ブロックリファレンス"では、Simulink ブロック全てのリファレンスを 提供します。

第 10 章の " モデル構築コマンド " では、MATLAB コマンドウィンドウまたは M-ファイルから、モデルを作成したり修正するために使用することができるコマン ドについて、リファレンス情報を提供します。

第11章の "Simulink デバッガ"では、Simulink モデルをデバックするための Simulink デバッガの使い方に関して解説します。

第 12 章の "Performance Tools" では、Simulink モデルの機能を向上させるための ツールである Simulink アクセラレータやオプションツールの使用方法を説明し ます。

付録 A の"モデルとブロックパラメータ"では、モデルパラメータとブロックパ ラメータのリストを提供します。この情報は、第10章で説明されている get_param コマンドおよび set_param コマンドで有効です。

付録 B の "モデルファイルフォーマット"では、モデル情報を格納するファイル のフォーマットについて説明します。

このマニュアルでは完全かつ最新の情報を提供できるよう最善を尽くしています が、いくつかの情報の一部が変更されている可能性もあります。最新のリリース 情報については、Simulink システムと一緒に提供される Known Software and Documentation Problems で確認してください。

関連プロダクト

The MathWorks は、Simulink を使って行うことができるタスクに関連するプロダクトを提供しています。

これらのプロダクトに関する情報は、以下を参照してください。

- ロードされている場合、または CD からドキュメントを読み込んでいる場合は
 そのプロダクトのオンラインドキュメント
- ・ The MathWorks Web サイト、www.mathworks.com の "products" セクション

新しい製品や機能を最新のものに更新するには、The Math Works の Web サイト www.mathworks.com をご覧ください。また、Simulink と互換性のあるサードパー ティ製品については、www.mathworks.com/products/connections/ をご覧ください。

注意 以下に示すツールボックスは、すべて MATLAB 環境を拡張する関数を含んでいます。ブロックセットは、すべて Simulink 環境を拡張するブロックを含んでいます。

プロダクト	説明
μ-Analysis and Synthesis Toolbox	最適化制御と構造化特異値を用いたロバスト制 御設計のためのツール
CDMA Reference Blockset	IS-95A ワイヤレス通信標準の設計とシミュレー ションのための Simulink ブロックライブラリ
Communications Blockset	通信系の物理レイヤのモデリングのための Simulink ブロックライブラリ
Communications Toolbox	通信系の物理レイヤのモデリングのための MATLAB 関数
Control System Toolbox	古典および現代制御システムの設計、解析、モ デリングのための対話的な環境

プロダクト	説明
Dials & Gauges Blockset	Simulink モデルの信号とパラメータのモニタリ ングと制御のためのグラフィカルな仮想計器
DSP Blockset	ディジタル信号処理系の設計、シミュレーショ ン、プロトタイプ用 Simulink ブロックライブラ リ
Fixed-Point Blockset	固定小数点アプリケーションのモデリング、シ ミュレーション、純粋な整数コードの自動生成 のための Simulink プロック
Frequency Domain System Identification Toolbox	周波数領域モデルの同定と実証のためのツール
Motorola DSP Developer's Kit	Motorola Suite56 TM DSP シミュレータをコールす る MEX- ファイルや S- ファンクションをプログ ラムするためのオブジェクト試行のインタ フェースを提供します。
Nonlinear Control Design (NCD) Blockset	システム設計のための時間領域ベースの最適化 手法を提供する Simulink プロックライプラリ。 ユーザ定義の時間領域の性能の制約に基づき、 パラメータを自動的に調整します。
Power System Blockset	電力システムの設計、解析、プロトタイプ用の Simulink ブロックライブラリ
Real-Time Windows Target	PC 上でリアルタイムで対話的に Simulink モデル を実行するためのツール
Real-Time Workshop [®]	.Simulink および Stateflow モデルからカスタマイ ズされたコードを生成するツール
Real-Time Workshop Ada Coder	Ada 95 コードを自動生成するためのツール。 Simulink モデルから直接コード生成し、様々な 環境でリアルタイムで実行できるプログラムを 自動的に構築します。

プロダクト	説明
Real-Time Workshop Production Coder	組み込み可能な製品コードを Simulink モデルか ら生成するためのコンポーネントが加えられま す。コ・シミュレーション内で生成されたコー ドや、コード生成インタフェースオプションを 検証するためのユーティリティや機能が含まれ ます。
Requirements Management Interface	このインタフェースは、開発サイクル全体にお いて、設計仕様 (要求) での変更を調整、追跡、 実現します。
Robust Control Toolbox	先進のロバスト多変数フィードバック制御系設 計のためのツール
Signal Processing Toolbox	アルゴリズム開発や、信号、線形システム解析、 時系列データモデリングのためのツール
Simulink Performance Tools	モデルの相違点、プロファイリングおよびシ ミュレーション性能を向上させるためのツール が含まれます。
Simulink Report Generator	MATLAB, Simulink, Stateflow の情報を多彩な出 力形式でドキュメント化するツール
Stateflow	複雑なリ・アクティブシステムのグラフィカル なモデリングとシミュレーションのためのツー ル
Stateflow Coder	Stateflow チャートから高効率の読み込み可能な C コードを生成するツール
System Identification Toolbox	ノイズのある時系列データから複雑なシステム の精度の高い簡素化されたモデルを作成するた めの対話的な環境

プロダクト	説明
Developer's Kit for Texas Instruments DSP	Texas Instruments (TI) C6701 評価モジュール (C6701 EVM) 上で、Simulink モデルを生成、 ターゲットとし、実行することができます。
xPC Target	Simulink ブロック線図に I/O ブロックを付け加 え、Real-Time Workshop を使ってコード生成し、 コードを xPC Target リアルタイムカーネルを起 動するほかの PC にダウンロードするツール。 xPC Target は、制御系および DSP システムのラ ピッドプロトタイピングおよび hardware-in-the-loop テストに対して理想的な ツールです。

クイックスタート

デモモデルの実行 デモの説明																			•	. 2	-2 -3
このデモが示すもの		•	•	•	•		•	•	•	•	•	•	•	•		•	•		•	· 2 · 2	-4 -4
その他の有効なデモ	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	. 2	,-5
簡単なモデルの作成.	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	. 2	-6
Simulinkの設定			•	•				•	•	•		•		•			•		•	.2-	15
Simulink Preferences.	•	•			•	•		•		•	•		•		•	•	•	•	•	2-	13

デモモデルの実行

Simulink が提供する興味深いデモプログラムで、住宅の熱力学をモデル化します。このデモを実行するためには、以下のステップに従ってください。

- 1 MATLAB を起動します。その方法がわからない場合は、MATLAB のドキュメ ントを参照してください。
- 2 MATLAB のコマンドウィンドウに thermo と入力して、デモモデルを実行しま す。このコマンドは Simulink を起動し、このモデルを含むモデルウィンドウ を作成します。



3 Thermo Plots とラベルされた Scope ブロックをダブルクリックしてください。

Scope ブロックには、Indoor vs. Outdoor Temp と Heat Cost (\$) とラベルされた 2 つのプロットが表示されます。

4 シミュレーションを開始するには、Simulation メニューをプルダウンし、Start コマンドを選択します(あるいは Windows の場合、Simulink ツールバーに配 置された Start ボタンを押します)。シミュレーションが実行されると、室内 と室外の温度が Indoor vs. Outdoor Temp の Scope ブロックに、累積暖房費出力 が Heat Cost (\$) の Scope ブロックに表示されます。

- 5 シミュレーションを終了するには、Simulation メニューから Stop コマンドを選択します(もしくは、ツールバーの Pause ボタンを押します)。モデルの他の 部分を調べたい場合は、2-4 ページの"試してみるべきこと"の指示を参照して ください。
- シミュレーションの実行が終了したら、File メニューから Close を選択してモデルを閉じます。

デモの説明

デモは、簡単なモデルを用いて住宅の熱力学をモデル化します。サーモスタット (自動温度調節器)は華氏 70度に設定され、外気温の影響を受けます。外気温 は、15度の振幅をもつ正弦波を 50度の基準温度に適用することによって変化し ます。これは、日常の温度変動をシミュレーションします。

モデルは、モデル図を単純化し、再利用可能なシステムを作成するために、サブ システムを使います。サブシステムは、Subsystem ブロックで表わされるブロッ クのグループです。このモデルには5つのサブシステムが含まれています。 Thermostat、House、そして Temp Convert が3つ(2つは華氏を摂氏に変換し、1 つは摂氏を華氏に変換します)です。

内外の温度が House サブシステムに取り入れられて、屋内温度を変更します。 House ブロックをダブルクリックすると、そのサブシステム内のブロックを見る ことができます。



House subsystem

Thermostat サブシステムは、サーモスタットの動作をモデル化し、暖房システムのオンとオフの切り替えを決定します。そのブロックをダブルクリックすると、そのサプシステム内のブロックを見ることができます。



Thermostat subsystem

内外の温度は、いずれも同じサブシステムによって華氏から摂氏に変換されま す。



Fahrenheit to Celsius conversion (F2C)

暖房がオンの場合、暖房費が計算され、Heat Cost (\$)の Thermo Plots Scope ブロック上に表示されます。屋内温度は、Indoor Tempの Scope ブロック上に表示されます。

試してみるべきこと

モデルが、異なるパラメータにどのように反応するかを知るために、試してみる ことがいくつかあります。

- 各 Scope ブロックには信号表示領域とコントロールボタンがあります。コントロールにより、表示された信号の範囲を選択したり、信号の一部を拡大したり、その他の役に立つ作業を実行したりすることができます。水平軸は時間を表し、垂直軸は信号値を表します。Scope ブロックの詳細については9-198ページの Scope を参照してください。
- (モデルの左上に)Set Pointの表示のある Constant ブロックは、希望する屋内温度 を設定します。このブロックをオープンし、シミュレーションが実行されて いる時に値を 80 度にリセットし、屋内の温度と暖房費がどのように変化する かを見てください。また、外気温 (Avg Outdoor Temp ブロック)を調整し、そ れがシミュレーションにどのように影響するかを見てください。
- Daily Temp Variation とラベルの付いた Sine Wave ブロックをオープンし、 Amplitude パラメータを変化させて、日常の温度変化を調整してください。

このデモが示すもの

このデモでは、モデル作成時に共通して使用されるいくつかの作業を示します。

- シミュレーションの実行には、パラメータの指定と Start コマンドによるシミュレーションの開始が含まれます。これらについては5-4ページの"メニューコマンドを用いたシミュレーションの実行"で詳しく説明します。
- ・ 関連ブロックの複合グループを単一ブロックにまとめることができます。これはサブシステムと呼ばれます。サブシステムの作成の詳細については 4-66ページの"サブシステムの作成"で説明します。
- 第7章の"マスクを使ったブロックのカスタマイズ"で詳しく説明するマスク機能を用いて、ブロックに対するアイコンをカスタマイズし、ダイアログボックスを設計することができます。thermoモデルでは、すべてのSubsystemブロックに、マスク機能を用いて作成したカスタマイズ済みのアイコンがあります。
- Scope ブロックは、実際のオシロスコープと同じようにグラフィック出力を表示します。Scope ブロックについては、9-198 ページの Scope で説明します。

その他の有効なデモ

他のデモは、モデル化の概念を示します。これらのデモは、Simulink ブロックラ イブラリウィンドウからアクセスできます。

1 MATLAB コマンドウィンドウに simulink3 と入力します。Simulink のブロック ライブラリウィンドウが表示されます。



 Demos アイコンをダブルクリックします。MATLAB Demo Window が表示され ます。このウィンドウには、有効な Simulink 機能を示すいくつかの興味深い サンプルモデルが含まれています。

簡単なモデルの作成

この例は、多くのモデル作成コマンドを用いてモデルを作成する方法と、独自の モデルを作成するための動作を示します。この節では、このモデルを作成するた めの方法を簡単に説明しています。すべての作業については、つぎの章で詳細に 説明します。

モデルは正弦波を積分し、結果をその正弦波と共に表示します。モデルのブロッ ク線図は、つぎのとおりです。



モデルを作成するために、MATLAB コマンドウィンドウで simulink とタイプイ ンしてください。Microsoft Windows では、Simulink ライブラリブラウザが現れ ます。



UNIX では、Simulink ライブラリウィンドウが現れます。



UNIX では、新しいモデルを作成するために、Simulink ライブラリウィンドウの File メニューの New サプメニューから Model を選択してください。Windows で 新しいモデルを作成するために、ライブラリブラウザのツールバーの New Model ボタンを選択してください。



b- untitled1	_ 🗆 ×
<u>File Edit View Simulation Format Tools</u>	
🛯 🖻 🖬 🎒 🐰 🖬 🖬 의 오 🕨 🗉	u 🛛 🛺
	1
	///

このモデルを作成するために、つぎの Simulink ブロックライブラリからモデル にブロックをコピーする必要があります。

- Sources ライブラリ (Sine Wave ブロック)
- Sinks ライブラリ (Scope ブロック)
- Continuous ライブラリ (the Integrator ブロック)
- ・ Signals & System ライブラリ (Mux ブロック)

ライブラリブラウザ (Windows のみ) を使って Sources ライブラリから、もしく は Sources ライブラリウィンドウ (UNIX と Windows) から Sine Wave ブロックを コピーできます。

ライブラリブラウザから Sine Wave ブロックをコピーするために、まずライブラ リブラウザのツリーを広げて Sources ライブラリのブロックを表示します。これ を実行するために、まず Simulink ノードをクリックし Sources ノードを表示し、 つぎに Sources ノードをクリックし、Sources ライブラリを表示します。最後に Sine Wave ブロックを選択するために、Sine Wave ノードをクリックします。以 上のことを実行した後のライブラリブラウザの様子を示しておきます。
ii Simulink Library Browser File Edit ⊻iew <u>H</u> elp			
□ □ □ Sine Wave: Output a sine wave.			
□ 💽 Simulink	M	Pulse Generator	Simuliak ライブラリ
2 Discrete 2 Functions & Tables 2 Math		Ramp	
	٨w	Random Number	
25 Sinks 25 Sources	М	Repeating Sequence	
È Simulink Extras ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓	0000 \$\$	Signal Generator	Sources ライブラリ
	\mathbb{A}	Sine Wave	
		Step	←── Sine Wave ブロック
	\square	Uniform Random Number	
Ready			

ここで、ブラウザから Sine Wave ノードをドラッグし、モデルウィンドウにド ロップします。Simulink は、ノードアイコンをドロップした場所に Sine Wave ブ ロックをコピーします。

Sources ライブラリウィンドウから Sine Wave ブロックをコピーするために、 Simulink ライブラリウィンドウの Sources アイコンをダブルクリックして Sources ウィンドウをオープンします (Windows では Simulink ライブラリウィンドウを オープンするためにライブラリブラウザで Simulink ノードを右クリックし、そ れから Open Library ボタンをクリックします)。Simulink は Sources ライブラリ ウィンドウを表示します。



ここで、Sources ウィンドウからモデルウィンドウに、Sine Wave ブロックをド ラッグします。



残りのブロックも同様の方法でそれぞれのライブラリからモデルウィンドウにコ ピーします。モデルウィンドウのある場所から他の場所にブロックを動かすため にブロックをドラッグします。ブロックを選択し、矢印キーを押すことにより、 ブロックの移動の微調整ができます。 すべてのブロックがモデルウィンドウにコピーされると、モデルはつぎのように 表示されます。



ブロックアイコンをよく見ると、Sine Wave ブロックの右側に角括弧 (>) があり、 Mux ブロックの左側には 2 つの角括弧があるのがわかります。ブロックから > 記号が出ている場合は*出力端子*です。記号がブロックを指している場合は*入力端 子*です。信号は出力端子から出て、接続線を通じて別のブロックの入力端子に入 ります。ブロックが接続されると、端子記号は消えます。



ここでブロックを結線します。Sine Wave ブロックを Mux ブロックの上部の入力 端子に結線します。ポインタの位置を Sine Wave ブロックの右側の出力端子に合 わせます。カーソルの形がクロスヘア (十文字)に変わります。



マウスボタンを押して、カーソルを Mux ブロックの最上部の入力端子に移動します。マウスボタンを押している間はラインは破線となり、カーソルを Mux ブロックに近づけるとカーソルの形が二重のクロスへアに変わるのがわかります。



ここで、マウスボタンを解除します。ブロックは結線されています。ポインタが アイコン内部にある間にマウスボタンを解除することによって、ラインをブロッ クに接続することもできます。こうすると、ラインはカーソルの位置にもっとも 近い入力端子に接続されます。



この章の始めのモデルをもう一度見ると、ほとんどのラインがブロックの出力端 子を他のブロックの入力端子に結線していることがわかります(2-6ページの"簡 単なモデルの作成"を参照)。ただし、1本のラインは*ライン*を別のブロックの入 力端子に結線しています。これは*分岐線*と呼ばれ、Sine Wave 出力を Integrator ブ ロックに接続し、Sine Wave ブロックから Mux ブロックに渡されるのと同じ信号 を伝搬します。

分岐線の描画は、先に引いたラインの描画とは少し異なります。既存のラインと の結線を行うには、つぎの手順に従ってください。

1 まず、ポインタの位置を Sine Wave ブロックと Mux ブロックとの間のライン上 に合わせます。



2 Ctrl キーを押しながらマウスボタンを押します。マウスボタンはそのままにして、ポインタを Integrator のブロック入力端子、または Integrator ブロックそのものにまでドラッグします。



3 マウスボタンを解除します。Simulink は、起点と Integrator ブロックの入力端子 との間にラインを引きます。



ブロック接続を終了します。終了後、モデルはつぎのようになります。



つぎに、Scope ブロックを開いてシミュレーション出力を表示します。Scope ウィンドウを開いたまま、10 秒間のシミュレーションを実行します。まず、 Simulation メニューから Simulation Parameters を選択して、シミュレーション パラメータを設定します。表示されるダイアログボックス上で、Stop time が 10.0(デフォルト値)に設定されていることに注意してください。

eter

OKボタンをクリックしてSimulation Parametersダイアログボックスを閉じます。 Simulink はパラメータを適用し、ダイアログボックスを閉じます。



Simulation メニューから Start を選択し、Scope ブロックの入力を追跡します。

シミュレーションは、Simulation Parameters ダイアログボックスで指定した終 了時間に達するか、Simulation メニューから Stop を選択すると停止します。

このモデルを保存するには、File メニューから Save を選択し、ファイル名と保存場所を入力します。そのファイルにはモデルの説明が含まれています。

Simulink と MATLAB を終了するには、File メニューから Exit MATLAB (Microsoft Windows システムの場合)、Quit MATLAB (UNIX システムの場合を選 択します。MATLAB コマンドウィンドウに quit と入力することもできます。 Simulink を終了しても MATLAB を終了したくない場合には、すべての Simulink ウィンドウのみを閉じてください。

この演習では、一般的に用いられるモデル作成作業のいくつかを実行する方法を 示しました。モデル作成やその他の作業の詳細については第4章の"モデルの作 成"で説明します。

Simulink の設定

MATLAB **Preferences** ダイアログボックスから、Simulink オプションのデフォルト設定を定義することができます。**Preferences** ダイアログボックスを表示するには、Simulinkの**File** メニューから **Preferences** を選択します。

Preferences		×
General Command Window Editor/Debugger Ourrent Directory Workspace Array Editor GUIDE Figure Copy Template Simulink Fonts Simulation	Simulink Preferences Window reuse: mixed Model Browser Show masked subsystems Show library links Browser initially visible Display Wide nonscalar lines Show port data types Callback tracing	
	OK Cancel Help	

Simulink Preferences

Preferences ダイアログボックスでは、つぎの Simulink 設定を行うことができます。

Window reuse

モデルのサブシステムを表示するときに、既に存在するウィンドウを使用するか、新しいウィンドウを開くかを指定します (4-68 ページの"ウィンドウの再利用"参照してください)。

Model Browser

モデルのオープン時にモデルブラウザを表示するかどうか、また、サブシステム から取り込まれたブロックやマスクされたサブシステムの内容をブラウザに表示 するかどうかを指定します (4-100 ページの " モデルブラウザ " を参照してください)。

Display

ブロック間の非スカラ結合の表示に幅広のラインを使用するかどうか、また、ブロックダイアグラムに端子データタイプを表示するかどうかを指定します (4-38 ページの "信号の表示オプションの設定"を参照)。

Callback tracing

モデルをシミュレーションするときに、Simulink が呼び出すモデルのコールバッ クを表示するかどうかを指定します (4-71 ページの"コールバックルーチンの使 用法"を参照してください)。

Simulink Fonts

ブロックやラインのラベルや図の注釈に使用するフォントを指定します。

Solver

シミュレーションソルバオプションを指定します(5-8ページの"Solverページ"を 参照してください)。

Workspace

シミュレーションモデルのワークスペースオプションを設定します (5-18 ページの "Workspace I/O ページ"を参照してください)。

Diagnostics

シミュレーションモデルの診断オプションを設定します (5-25 ページの "Diagnostics ページ"を参照してください)。

3

Simulink の機能

Simulink とは	3-2
ダイナミックシステムのモデリング	3-3
ブロック線図	3-3
$\vec{\mathcal{J}}$	3-3
状態	3-4
システム関数	3-4
ブロックパラメータ	3-5
連続ブロックと離散ブロック	3-6
サブシステム	3-6
h = h = h = h = h = h = h = h = h = h =	3-6
信号	3-7
データタイプ	3-7
ソルバ	3-8
ダイナミックシステムのシミュレーション	3-9
モデルの初期化フェーズ	3-9
モデル実行フェーズ	. 3-10
時間ステップ毎の処理	. 3-10
ブロックのアップデート順の決定	3-11
Atomic サブシステムと仮想サブシステム	3-13
ソルバ	3-13
ゼロクロッシングの検出	3-14
代数ループ	3-18
	. 510
離散システムのモデル化とシミュレーション	3-23
離散ブロック	. 3-23
サンプル時間	. 3-23
純粋な離散システム	. 3-23
マルチレートシステム	3-24
離散システムのステップサイズの決定	3-24
サンプル時間の伝播	3-26
不变定数	3-27
うんん うちょう ちょう ちょう ちょう ちょう ちょう ちょう ちょう うちょう	3 28

Simulink とは

Simulink は、ダイナミックシステム、つまり出力と状態が時間と共に変化するシ ステムのモデル化、シミュレーション、解析を行うためのソフトウェアパッケー ジです。Simulinkを使って、電気回路、ショックアブソーバ、ブレーキシステム を含む多くの電気、機械、熱力学システムのような広範囲な実世界のシステムの 挙動を探求することができます。

ダイナミックシステムのシミュレーションは、Simulinkを使うと2ステップのプロセスです。最初に、Simulinkのモデルエディタを使ってシステムの入力、状態、出力の中から時間依存の数学的な関係をグラフィカルに記述するシステムのモデルを作成します。(3-3ページの"ダイナミックシステムのモデリング"を参照)。その次に、Simulinkを使って指定した時間範囲でシステムをシミュレートします。Simulinkは、モデルが提供する情報を使ってシミュレーションを行います(3-9ページの"ダイナミックシステムのシミュレーション"を参照)。

ダイナミックシステムのモデリング

Simulink では、ブロック線図を使ってダイナミックシステムをモデリングしま す。Simulink は、紙と鉛筆で図を描くのと同様の point-and-click 法を使い、マウ スを使ってスクリーン上にブロック線図の描画を行うことができるグラフィカル エディタを提供しています。(第4章の"モデルの作成")。Simulink は、実世界ダ イナミックシステムを仮想的にモデリングするために様々な組み合わせて利用で きる標準ブロックの拡張ライブラリも提供します(第9章の"ブロックリファレ ンス"を参照)。

ブロック線図

ブロック線図は、ラインによって接続されるブロックと呼ばれるシンボルで構成 されます。各ブロックは、連続的(continuous ブロック)または特定の時間点 (discrete ブロック)で出力を生成するダイナミックシステムを表わします。ライ ンは、ブロック入力とブロック出力の接続を表わします。ブロック線図内の各ブ ロックは、特定のタイプのブロックです。ブロックのタイプは、ブロックの出力 と、入力、状態、時間との関係を決定します。ブロック線図は、システムのモデ リングに必要な任意のタイプ、任意数のブロックを含みます。

注意 The MathWorks web site の MATLAB Based Books ページには、一般的なブロック線図の利用や、特にダイナミックシステムのモデリングを議論するテキストが含まれます。

ブロック

ブロックは、シミュレート方法がわかっている基本のダイナミックシステムを表 わします。ブロックは、入力、状態、出力のうちの1つ、または複数を含みま す。



ブロックの出力は、時間の関数で、ブロックの入力は状態です。ブロックの出力 と入力、状態、時間を関連付ける関数は、ブロックのタイプにより異なります。

状態

ブロックは、状態をもつことができます。状態は、ブロックの出力を決定する 変数で、現在の値は過去の状態や入力の関数となります。状態をもつブロック は、状態の前の値を格納して、カレント値を計算する必要があります。そのた め、状態は、持続すると言えます。状態をもつブロックは、状態のカレント値を 計算するために状態または入力の前の値を格納する必要があるため、メモリをも つと言えます。

Simulink Integrator ブロックは、状態をもつブロックの例です。Integrator ブロッ クは、シミュレーションの開始時からカレントの時間までの入力信号の積分を出 力します。カレントの時間ステップでの積分は、Integrator ブロックの入力の履 歴により異なります。そのため、積分は Integrator ブロックの状態で、実際に唯 ーの状態です。状態をもつブロックのその他の例は、Simulink Memory ブロック です。Memory ブロックは、カレントのシミュレーション時の入力値を格納し、 後でそれらを出力します。Memory ブロックの状態は、入力の前の値です。

Simulink Gain ブロックは、状態のないブロックの例です。Gain ブロックは、入 力信号にゲインと呼ばれる定数を乗算して出力します。Gain ブロックの出力は、 完全に、変化しない入力のカレント値とゲインの値によって決定されます。プ ロックは、そのため、状態をもちません。状態をもたないブロックのその他の例 は、Sum および Product ブロックです。これらのブロックの出力は、純粋にそれ らの入力のカレント値の関数です (Sum の場合は和で、Product の場合は積)。そ のため、これらのブロックは状態をもちません。

システム関数

各 Simulink ブロックのタイプは、入力、状態、出力の時間依存の関係を指定す るシステム関数に関連しています。システム関数は、以下のものです。

- ・ 出力関数 fo、システムの出力と入力、状態、時間を関連付けます。
- アップデート関数 fu,、システムの離散状態の将来の値とカレントの時間、入力、状態を関連付けます。
- 微係数関数、システムの連続状態の微係数と、時間とブロックの状態と入力の現在の値を関連付けます。

システム関数は、シンボルとしてつぎのように表わされます。

 $y = f_o(t, x, u)$ Output function $x_{d_{k+1}} = f_u(t, x, u)$ Update function $x'_c = f_d(t, x, u)$ Derivative function where $x = \begin{bmatrix} x_c \\ x_{d_k} \end{bmatrix}$

ここで、t はカレントの時間、x はブロックの状態、u はブロックの入力、y はブ ロックの出力、xd はブロックの離散微係数、x'c はブロックの連続状態の微係数 です。シミュレーション中に、Simulink はシステム関数を呼び出し、システムの 状態と出力の値を計算します。

ブロックパラメータ

多くの標準ブロックの重要な特性は、パラメータ化されることです。たとえば、 Simulink の標準の Gain ブロックのゲインは、パラメータです。パラメータ化さ れたプロックは、モデルの編集時やシミュレート時にパラメータの値を設定でき るブロックダイアログをもちます。MATLAB 表現を使って、パラメータ値を指 定することができます。Simulink は、シミュレーションの実行前に表現を評価し ます。シミュレーション中にパラメータ値を変更することができます。これによ り、パラメータに対して最適の値を対話的に決定することができます。

パラメータ化されたブロックは、同じブロックのファミリを効率的に表わしま す。たとえば、モデルの作成時に、各 Gain ブロックのゲインパラメータを個別 に設定できるので、各ブロックは異なるふるまいを行います。各標準ブロックが ブロックのファミリを表わすことができるので、ブロックのパラメータ化は、 Simulink の標準ライブラリのモデリング能力を大きく向上させます。

調整可能なパラメータ

ブロックパラメータのほとんどは、調整することができます。 調整可能なパラ メータは、Simulink がモデルを実行している間にパラメータを変更することがで きます。たとえば、Gain ブロックのゲインパラメータは調整可能です。シミュ レーションが実行されている間に、ブロックのゲインを変更することができま す。シミュレーション中にパラメータの調整が行えないパラメータについては、 シミュレーション中にパラメータの設定が出来なくなります。また、ユーザが指 定したパラメータ以外のすべてのパラメータを調整不可能に設定することもでき ます。この設定により、大規模モデルの実行速度が向上し、より高速のモデル コードを生成することが可能です。詳細は、5-29 ページの"モデルパラメータの 設定"を参照してください。

連続ブロックと離散ブロック

Simulink の標準ブロックセットには、連続ブロックと離散ブロックが含まれま す。連続ブロックは、連続的に変化する入力に連続的に応答します。反対に、離 散ブロックは、ブロックのサンプル時間と呼ばれる一定間隔の積分刻みでのみ入 力の変更に応答します。離散ブロックは、連続する時間で出力を保持します。離 散ブロックは、サンプルレートを指定できるサンプル時間パラメータを含みま す。連続ブロックの例は、Constant ブロックと、Simulink の Continuous ブロック ライブラリのブロックです。離散ブロックの例は、Discrete Pulse Generator と Discrete ブロックライブラリのブロックです。

多くの Simulink ブロック、たとえば、Gain ブロックは、連続ブロックまたは離 散ブロックに接続しているかどうかによって、連続または離散になることができ ます。離散または連続になることができるブロックは、暗示的なサンプルレート をもちます。暗示的なサンプル時間は、ブロックの入力が連続である場合は連続 です。暗示的なサンプル時間は、すべての入力サンプル時間が最短の刻みの倍数 である場合、最短の入力サンプル時間と等しくなります。そうでない場合、入力 サンプル時間は、入力の基本サンプル時間に等しくなります。ここで、基本サ ンプル時間は、サンプル時間の集合の最大公約数として定義されます。

Simulink は、オプションで、ブロック線図を色分けして、含んでいるブロックの サンプル時間を識別することができます。例:黒(連続)、マジェンタ(定数)、 黄(ハイブリッド)、赤(最も速い離散)。詳細は、3-28 ページの"連続および離 散混在システム"を参照してください。

サブシステム

Simulink は、複雑なシステムをブロック線図によって表わされた相互接続したサ ブシステムの集合としてモデリングすることができます。Simulink の Subsystem ブロックと Simulink モデルエディタを使ってサプシステムを作成します。階層 的なモデルを作成するために、任意のレベルでサプシステムにサプシステムを組 み込むことができます。トリガ入力またはイネーブル入力で遷移が生じるときに のみ実行される、条件付きで実行されるサプシステムを作成できます(第8章の "条件付きで実行されるサブシステム."を参照)。

カスタムブロック

Simulink を使って、モデルで利用することができるカスタムブロックのライブラ リを作成することができます。グラフィカルに、またプログラマブルにカスタム ブロックを作成できます。グラフィカルにカスタムブロックを作成するには、ブ ロックの挙動を表わすブロック線図を描画し、このブロック線図を Simulink の Subsystem ブロック内に wrap し、Simulink のブロックのマスク機能を使ってブ ロックにパラメータダイアログを設定します。ブロックをプログラマブルに作成 するには、ブロックのシステム関数を含む M-ファイルまたは MEX-ファイルを 作成します (Writing S-Functions を参照)。結果のファイルは、S-ファンクション と呼ばれます。その後その S-ファンクションをモデル内の Simulink の S-function ブロックと関連付けます。S-function ブロックを Subsystem ブロック内 に設定し、パラメータダイアログを Subsystem ブロックに追加することにより、 パラメータダイアログを S-function ブロックに追加できます。

信号

Simulink は、ブロックの出力値を参照するために用語 signal を使います。 Simulink を使って、信号名、データタイプ(例、8 ビット、16 ビット、32 ビット 整数)、数値タイプ(実数または複素数)、大きさを(1 次元または2 次元配列) 含む多くの信号の属性を指定することができます。多くのブロックは、任意の データタイプまたは数値タイプ、大きさの信号を受け取り、出力できます。他の ブロックは、扱うことができる信号の属性に制限があります。

データタイプ

データタイプという言葉は、コンピュータシステムの内部表現を示しています。 Simulink でも、MATLAB がサポートする組込みのデータタイプ、たとえば、 int8, double や boolean(4-45 ページの"データタイプの機能"参照)といったパラ メータや信号を扱うことができます。さらに Simulink は、つぎの2つの Simulink 固有のデータタイプを定義します。

- Simulink.Parameter
- Simulink.Signal

これら Simulink 固有のデータタイプは、int32 にような汎用的な数値タイプでは 扱えないような Simulink 固有の情報を取得するためのタイプです。Simulink で は、Simulink モデル内のパラメータや信号と同じように、データオブジェクトと 呼ばれる Simulink データタイプを作成したり使用したりすることができます。

これらの Simulink データタイプを両方使用することで、モデル固有の情報を取得するためのデータタイプを作成することができます。

注意 Simulink ユーザインタフェースやマニュアルでも、組込みの MATALB タ イプのような非拡張データタイプと区別するために、Simulink データタイプをク ラスと呼んでいます。

ソルバ

Simulink モデルは、モデルの連続状態での時間微分係数を定義しますが、状態自 身の値ではありません。そこで、システムをシミュレーションするときは、シス テムの状態の微分係数を数値積分して連続状態を計算しなければなりません。汎 用的な数値積分テクニックとして、特有の長所を持った様々な方法が存在しま す。Simulink は、常微分方程式 (ODE) ソルバと呼ばれる、最も安定で効果的で、 さらに誤差の少ない数値積分法を実行します。ユーザは、シミュレーションを実 行するときにモデルに使うソルバを指定することができます。

ダイナミックシステムのシミュレーション

ダイナミックシステムのシミュレーションは、システムのモデルが提供する情報 を使って、システムの状態と出力を時間領域で計算するプロセスを参照します。 Simulink は、システムのモデルをオープンして、モデルエディタの Simulation メ ニューから Start を選択すると、システムをシミュレートします。

システムのシミュレーションは、モデルの初期化と、モデルの実行の2つの フェーズで発生します。

モデルの初期化フェーズ

初期化フェーズ中に、Simulink は、

- 1 モデルのブロックパラメータ表現を評価し、値を決定します。
- 仮想サブシステムが含んでいるモデルによって仮想サブシステムを置き換えることにより、モデルの階層を平坦にします。(3-13 ページの "Atomic サブシステムと仮想サブシステム"を参照)。
- 3 実行フェーズ中にアップデートする必要がある順番にブロックを並べ替えます(3-11ページの"ブロックのアップデート順の決定"を参照)。
- 4 モデルによって明示的に指定されていない信号の属性(例、名前、データタイ プ、数値タイプ、大きさ)を決定し、各ブロックが、入力に接続されている 出力を受け取ることができることをチェックします。

Simulink は、属性の伝播と呼ばれるプロセスを使って、指定されていない属性を決定します。このプロセスは、source 信号から接続しているブロックの出力への属性の伝播を伴います。

- 5 サンプル時間が明示的に指定されていないモデル内のすべてのブロックのサンプル時間を決定します。
- 6 各ブロックの状態と出力のカレント値を格納するためのメモリを割り当て、 初期化します。

モデル実行フェーズ

シミュレーションがモデル実行フェーズに入っています。このフェーズでは、 Simulink は、引き続いてモデルが提供する情報を使って、シミュレーション開始 時間から終了時間までの区間でシステムの状態と出力を計算します。状態と出力 が計算される連続する時間点は、時間ステップと呼ばれます。ステップ間の時間 の長さは、ステップサイズと呼ばれます。ステップサイズと呼ばれます。ステッ プサイズは、システムの連続状態の計算に用いるソルバのタイプ(3-13 ページの "ソルバ"を参照)、システムの基本サンプル時間(3-23 ページの"離散システム のモデル化とシミュレーション"を参照)、システムの連続状態が不連続性をもつ かどうかに依存します(3-14 ページの"ゼロクロッシングの検出"を参照)。

シミュレーションの開始時に、モデルは、シミュレートするシステムの初期状態 と初期出力を指定します。各ステップで、Simulink はシステムの入力、状態、出 力に対して新しい値を計算し、モデルをアップデートして計算値を反映させま す。シミュレーションの終了時に、モデルはシステムの入力、状態、出力の最終 値を反映させます。Simulink は、データの表示とロギングを行うブロックを提供 しています。モデルにこれらのブロックを含めることにより、中間結果を表示ま たはロギングできます。

時間ステップ毎の処理

各時間ステップで Simulink は、

1 並び替えられた順番でモデルのブロックの出力をアップデートします (3-11 ページの "ブロックのアップデート順の決定 "を参照)。

Simulink は、ブロックの出力関数を呼び出してブロックの出力を計算します。 Simulink は、出力関数がブロックの出力の計算のために、カレントの時間と ブロックの入力および状態を引数として必要とするため、これらを出力関数 に渡します。Simulink は、カレントのステップがブロックのサンプル時間の 定数倍である場合にのみ、離散ブロックの出力をアップデートします。

2 並び替えられた順番でモデルのブロックの状態をアップデートします。

Simulink は、離散状態アップデート関数を呼び出してブロックの離散状態を 計算します。Simulink は、連続状態の時間微係数を数値積分して、ブロック の連続状態を計算します。ブロックの連続状態の微分関数を呼び出して、状 態の時間微係数を計算します。 3 オプションでブロックの連続状態の不連続性をチェックします。

Simulink は、ゼロクロッシング検出と呼ばれる手法を使って、連続状態の不 連続性を検出します。詳細は 3-14 ページの "ゼロクロッシングの検出"を参照 してください。

4 つぎの時間ステップに対する時間を計算します。

Simulink は、シミュレーション終了時間に達するまで、1から4を繰り返します。

ブロックのアップデート順の決定

シミュレーション中に、Simulink は、モデルのブロックの状態と出力を時間ス テップ毎に1回アップデートします。そのため、ブロックがアップデートされる 順番は、結果の正当性にとって重要です。特に、カレントの時間ステップでブ ロックの出力が入力の関数である場合、ブロックは、入力が接続しているブロッ クの後にアップデートされなければなりません。そうでない場合は、ブロックの 出力は不正になります。ブロックがモデルファイルに格納される順番は、必ずし もシミュレーション中にアップデートされる必要がある順番ではありません。そ の結果、Simulink はモデルの初期化フェーズ中にブロックを正しい順番に並べ替 えます。

Direct Feedthrough プロック

有効なアップデートの順番を作成するために、Simulink は、出力と入力との関係 にしたがって、ブロックをカテゴリ分けします。カレントの出力がカレントの入 力に依存するブロックは、*direct feedthrough* ブロックと呼ばれます。それ以外の すべてのブロックは、nondirect-feedthrough ブロックと呼ばれます。

direct-feedthrough ブロックの例には、Gain, Product, Sum ブロックが含まれます。 nondirect-feedthrough ブロックの例には、Integrator ブロック(出力は純粋に状態 の関数です)、Constant ブロック(入力をもちません)、Memory ブロック(出力 が前の時間ステップの入力に依存します)が含まれます。

ブロックの並べ替えの規則

Simulink は、つぎの基本的なアップデートの規則を使って、ブロックを並べ替えます。

各ブロックは、接続している direct-feedthrough ブロックの前にアップデートされなければなりません。

この規則は、direct-feedthrough ブロックへの入力がアップデート時に有効であることを保証します。

 Nondirect-feedthrough ブロックは、接続している direct-feedthrough ブロックの前 に更新される限り、任意の順番でアップデートできます。

この規則は、アップデートリストの先頭にすべての nondirect-feedthrough ブロックを任意の順番で置くことによって満足されます。そのため、Simulinkは、並べ替え処理中に nondirect-feedthrough ブロックを無視できます。

これらの規則の適用の結果は、nondirect-feedthrough ブロックが順番を特定せず にリストの先頭にあり、そのつぎに direct-feedthrough ブロックが接続するブロッ クに対して有効な入力を与えるような順番で現れるアップデートリストとなりま す。

並べ替え処理中に、Simulink は代数ループ、つまり、direct-feedthrough ブロック の出力が直接または間接的に入力と接続される信号ループの発生をチェックして フラグを付けます。そのようなループは、Simulink は出力を計算するために direct-feedthrough ブロックの入力を必要とするため、外見では deadlock condition を作成します。しかし、代数ループは、ブロックの入力と出力が未知である連立 代数方程式を表わすことができます。さらに、これらの方程式は、各時間ステッ プで有効な解をもつことができます。従って、Simulink は、実際に direct-feedthrough ブロックを含むループが代数方程式を表わすと仮定し、ブロッ クがシミュレーション中のアップデートされるたびに方程式を解こうとします。 詳細は、3-18 ページの"代数ループ"を参照してください。

ブロックプライオリティ

Simulink では、ブロックの処理順序に優先順位を付けることが可能です(4-20 ページの"ブロックプライオリティの割り当て"を参照)。優先順位の高いブロッ クが、低いブロックの前に実行されます。ただし、その優先順位がブロックの ソートルールに従っている場合のみ、その順序が守られます。

Atomic サブシステムと仮想サブシステム

サブシステムは、virtual または atomic です。Simulink は、ブロックのアップデートの順番の決定時に、仮想サブシステムの境界を無視します。反対に、Simulink は、つぎのブロックに移動する前に、atomic サブシステム内のすべてのブロックを実行します。条件付きで実行されるサブシステムは、atomic です。条件なしで実行されるサブシステムは、デフォルトで仮想です。しかし、atomic として指定することができます。これは、それ以外のプロックが完全に実行される前にサブシステムが実行されることを保証する必要がある場合に有効です。

ソルバ

Simulink は、ブロックの連続状態のカレント値を、状態の微係数を数値積分する ことによって計算します。数値積分のタスクは、ソルバと呼ばれる Simulink コ ンポーネントによって行われます。Simulink を使って、モデルのシミュレーショ ンに用いるソルバを選択できます。Simulink が提供するソルバは、固定ステップ ソルバと可変ステップソルバの2つのクラスに分けられます。

固定ステップソルバ

固定ステップソルバは、シミュレーションの時間区間を、ステップと呼ばれる一 定間隔の整数倍に分割します。そのため、初期推定から開始すると、各時間ス テップで、固定ステップソルバは変数のカレント値と微係数のカレント値から、 つぎの時間ステップでのシステムの状態変数の値を計算します。推定の精度は、 *ステップサイズ、*つまり連続する時間ステップ間に時間に依存します。一般に、 小さいステップサイズは、正確なシミュレーションを行いますが、システムの状 態の計算に多くのステップが必要なため、実行時間が長くなります。

可変ステップソルバ

可変ステップソルバは、指定したレベルの精度を満たすために、ステップサイズ をダイナミックに変化させます。そのようなソルバは、状態変数がゆっくりと変 化するときには、ステップサイズを大きくし(状態微係数の大きさで示されるよ うに)、状態変数が急速に変化するときには、ステップサイズを小さくします。 可変ステップソルバは、アプリケーションによって実行速度を犠牲にすることな く、精度の良い結果を得ることができます。

メジャーステップとマイナーステップ

ソルバの中には、シミュレーション時間をメジャーステップとマイナーステップ に細かく分けるものがあります。マイナー時間ステップは、メジャー時間ステッ プの下位区分を表わします。ソルバは、各メジャー時間ステップでの結果を生成 します。マイナー時間ステップでの結果を使ってメジャー時間ステップでの結果 の精度を改良します。

ゼロクロッシングの検出

ダイナミックシステムのシミュレーション時に、Simulink は、ゼロクロッシング の検出として知られる手法を使って、各時間ステップでシステムの状態変数の不 連続性をチェックします。Simulink がカレントの時間ステップ内で不連続性を検 出した場合は、不連続性が生じる正確な時間を決定し、その前後に付加的な時間 ステップをとります。この節では、ゼロクロッシングの検出がなぜ重要かを、ま たその機能について説明します。

状態変数の不連続性は、ダイナミックシステムの展開において意味のあるイベン トと同時に起こることがあります。たとえば、ボールが床で跳ねるときにその位 置で不連続性を伴います。不連続性は、ダイナミックシステムでの意味のある変 化を表わすので、不連続点で正確にシミュレーションすることが重要です。そう でない場合は、シミュレーションは、調査中のシステムの挙動に関して間違った 結論になることがあります。たとえば、跳ねるボールのシミュレーションを考え ます。ボールが床と接触する点がシミュレーションステップ間である場合は、シ ミュレーションされたボールは空中の反対の位置に現れます。これは、跳ねる ボールの物理現象に関して間違った結果を導きます。

このような結果のミスリードを防ぐために、シミュレーションステップが不連続 点において生じることが重要です。シミュレーション時間の決定について純粋に ソルバに依存するシミュレータは、この要求を十分に満足しません。たとえば、 固定ステップソルバを考えます。固定ステップソルバは、固定ステップサイズの 整数倍において状態変数の値を計算します。しかし、不連続点がステップサイズ の整数倍であるという保証はありません。不連続点をヒットする可能性を増やす ためにステップサイズを減らすことはできますが、これにより実行時間は大幅に 増加します。

可変ステップソルバを利用すれば解が求まるように思えます。可変ステップソル バは、変数がゆっくり変化するときにはステップサイズを増加させ、急速に変化 するときには減少させて、ステップサイズをダイナミックに調整します。不連続 点付近で、変数は特に急速に変化します。そのため、理論では、可変ステップソ ルバは不連続点を正確にヒットできます。問題は、不連続点を正確に決定するた めには、可変ステップソルバは小さいステップを多く取る必要があり、シミュ レーションが遅くなることです。

ゼロクロッシング検出機能

Simulink は、ゼロクロッシングの検出として知られる手法を使って、この問題に 対処します。この手法を使うと、ブロックは Simulink でゼロクロッシング変数 を登録できます。それらは、不連続点をもつ状態変数の関数です。ゼロクロッシ ング関数は、対応する不連続が生じるとき正または負の値からゼロを渡します。 シミュレーションステップの最後に、Simulink は、ゼロクロッシング変数を登録 したブロックに、変数をアップデートすることを確認します。その後、Simulink は最終ステップ以降に変数の符号が変化したかどうかをチェックします。そのよ うな変化は、不連続がカレントの時間ステップで生じたことを示します。

ゼロクロッシングが検出された場合、Simulink は符号が変化した変数の前の値と カレント値の間で内挿を行い、ゼロクロッシングの時間(例、不連続点)を推定 します。Simulink は、各ゼロクロッシングを順番に進めます。この方法で、 Simulink は状態変数が未定義である不連続点でのシミュレーションを防ぎます。

ゼロクロッシングの検出を使って、Simulink は過度に小さいステップサイズに並 べ替えを行わずに正確に不連続点をシミュレートします。多くの Simulink ブ ロックは、ゼロクロッシング検出をサポートします。結果は、不連続をもつシス テムを含むすべてのシステムのシミュレーションが高速で正確です。

実行の詳細

ゼロクロッシングを利用する Simulink ブロックの例は、Saturation ブロックです。 ゼロクロッシングは、Saturation ブロックでつぎのような状態イベントを検出し ます。

- 入力信号が上限に到達する。
- 入力信号が上限から離れる。
- 入力信号が下限に到達する。
- 入力信号が下限から離れる。

それ自身の状態イベントを定義する Simulink ブロックは、*厳密なゼロクロッシング*をもっていると考えられます。ゼロクロッシングイベントを明示的に知らせる必要がある場合には、Hit Crossing ブロックを使ってください。ゼロクロッシングを組み込んだブロックのリストについては、3-17 ページの"ゼロクロッシングをもつブロック"を参照してください。

状態イベントの検出は、内部のゼロクロッシング信号の構成によって異なりま す。この信号をブロック線図から利用することはできません。Saturation ブロッ クの場合、上限に対するゼロクロッシングを検出するために使われる信号は、 zcSignal = UpperLimit - u です、ここで、u は入力信号です。 ゼロクロッシング信号は方向属性をもっており、つぎのような値をとります。

- rising ゼロクロッシングは、信号がゼロから負に向かう瞬間またはゼロを通 過する瞬間、あるいはゼロから正になる瞬間で発生します。
- *falling* ゼロクロッシングは、信号が正からゼロに向かう瞬間またはゼロを通過する瞬間、あるいはゼロから負になる瞬間で発生します。
- *either* ゼロクロッシングは、rising と falling のいずれかの条件が発生した場合 に発生します。

Saturation ブロックの上限に対して、ゼロクロッシングの方向は、*either* です。これにより、同じゼロクロッシング信号を使って飽和イベントの切り替えを検出することができます。

誤差許容値が大きすぎる場合、Simulink は、ゼロクロッシングを検出できない可 能性があります。たとえば、ゼロクロッシングがある1つの時間ステップ内に生 じ、ステップの開始時と終了時で値の符号変換を示すことがないような場合を想 定してみましょう。このような場合、ソルバはゼロクロッシング部を検出できず に、その位置を横切ってしまいます。

つぎの図は、ゼロを横切る信号を示しています。最初の例では、積分はこのイベントを "飛び越えて " しまします。2 番目の例では、ソルバはこのイベントを検出します。



最初の例のような状況が起きる可能性がある場合、ソルバが十分小さなステップ をとるように、誤差許容値を小さくしてください。詳細は、5-13 ページの "Error Tolerances" を参照してください。

注意 Refine オプション (5-16 ページの"リファインファクタ"を参照)の使用は、 間違ったゼロクロッシングの位置決めに利用できません。最大ステップサイズや 出力時間を変更してください。

警告

不連続点に関して高周波変動(チャタリング)を示すモデルを作成することがで きます。たとえば質量のないバネのようなシステムは、一般に物理的には実現不 可能です。チャタリングによってゼロクロッシングは繰り返し検出されるので、 シミュレーションのステップサイズはきわめて小さくなり、実際上シミュレー ションは停止します。

このような挙動がモデルに適用される恐れがある場合は、Simulation Parameters ダイアログボックスの Diagnostics ページ上の Disable zero crossing detection チェックボックス(5-31 ページの"ゼロクロッシングの検知"を参照)を選択す ることにより、ゼロクロッシングを無効にすることができます。ゼロクロッシン グ検出を無効にすることで、この問題の徴候を緩和することができますが、ゼロ クロッシング検出が提供する精度向上の利点を利用することができなくなりま す。より良い解法は、モデルの根本的問題の原因を見つけることです。

ゼロクロッシングをもつブロック

ブロック	ゼロクロッシングの詳細
Abs	1 つ:正方向と負方向において、入力信号がゼロを横切る瞬 間を検出。
Backlash	2つ:1つは上限しきい値に到達する瞬間を検出し、もう1つ は下限しきい値に到達する瞬間を検出。
Dead Zone	2 つ:1 つは不感帯に入る瞬間(入力信号から下限値を差し引 いたもの)を検出し、もう1 つは不感帯から出る瞬間(入力 信号から上限値を差し引いたもの)を検出。
Hit Crossing	1 つ : 入力がしきい値を横切る瞬間を検出。これらのゼロク ロッシングは、Simulation Parameters ダイアログボックスの Disable zero crossing detection チェックボックスの影響を受け ません。
Integrator	リセット端子がある場合、リセットが生じる瞬間を検出。出 力に制限がある場合、3つのゼロクロッシングが存在します。 1つは、飽和上限に到達する瞬間を検出し、1つは飽和下限 に到達する瞬間を検出し、もう1つは飽和状態でなくなる瞬 間を検出します。

プロック	ゼロクロッシングの詳細 (Continued)
MinMax	1 つ : 出力ベクトルの各要素に対して、入力信号が新しい最 小値または最大値となる瞬間を検出。
Relay	1 つ : リレーがオフの場合、スイッチがオンになる時点を検 出。リレーがオンの場合、スイッチがオフになる瞬間を検 出。
Relational Operator	1 つ : 出力が変化する瞬間を検出。
Saturation	2 つ : 1 つは上限に到達するかまたは上限を離れる瞬間を検出 し、1 つは下限に到達するかまたは下限を離れる瞬間を検出。
Sign	1 つ : 入力が零点を横切る瞬間を検出。
Step	1つ:ステップ時間を検出。
Subsystem	条件付きで実行されるサブシステムに対して、1 つはイネー ブル端子が存在する場合の検出、もう1 つはトリガ端子が存 在する場合の検出。
Switch	1 つ:スイッチ条件が発生する瞬間を検出。

代数ループ

いくつかの Simulink ブロックは、*直接フィードスルー*を伴う入力端子をもって います。これは、これらの入力端子のブロックに入る信号の値を知らなければ、 これらのブロックの出力を計算することができないことを意味します。直接 フィードスルー入力をもつブロックに対して、いくつかの例があります。

- ・ Elementary Math ブロック
- Gain ブロック
- Integrator ブロックの初期条件端子
- Product ブロック
- ・ 非ゼロの D 行列が存在する場合の State-Space ブロック
- Sum ブロック
- ・ 分子と分母が同じ次数の場合の Transfer Fcn ブロック
- ・ 零点と極の数が等しい場合の Zero-Pole ブロック

ブロックに直接フィードスルーがあるかどうかを確認するには、第9章の"ブロックリファレンス、"のブロックを説明する特性表を参照してください。

代数ループは、直接フィードスルーをもつ入力端子が同じブロックの出力と直接接続されるか、または直接フィードスルーをもつ他のブロックを通るフィードバック経路を通じて接続される場合に発生します(この一般ルールの例外については、3-20ページの"非代数直接フィードスルーループ"を参照してください)。例として、つぎの簡単なスカラループがあります。



数学的にこのループは、Sum ブロックの出力は代数状態 z で、これは最初の入力 u から z を差し引いたもの(すなわち z = u - z)に等しくなるように制約されてい ることを意味しています。この単純ループの解は z = u/2 ですが、ほとんどの代 数ループは検査によって解くことはできません。ベクトル代数ループは、つぎの モデルに示すように、複数の代数状態変数 z1, z2 などを使って容易に作成するこ とができます。



Algebraic Constraint ブロック (Algebraic Constraint を参照)は、代数方程式をモデ ル化し、初期推定値を指定するのに便利です。Algebraic Constraint ブロックは、 その入力信号 F(z) をゼロに制約し、代数状態 z を出力します。このブロックは、 入力でゼロを生成するのに必要な値を出力します。出力はいくつかのフィード バック経路を通って入力に影響しなければなりません。ブロックのダイアログ ボックスで代数状態値の初期推定値を指定することにより、代数ループソルバの 効率を向上させることができます。 スカラ代数ループは、スカラ代数方程式または F(z) = 0 形式の制約を表します。 ここで、z はループ内のブロックの1つの出力であり、関数 F はループ内の他の ブロックを通ってそのブロックの入力に至るまでのフィードバック経路からなり ます。前のページに示した簡単な1 ブロックの例では、F(z) = z - (u - z)となりま す。上に示すベクトルループの例では、方程式はつぎのとおりです。

$$z^2 + z^2 - 1 = 0$$

$$z^2 - z^2 - 1 = 0$$

代数ループは、モデルに代数的な制約 F(z) = 0 が含まれるときに生じます。この 制約は、モデル化しようとするシステムの物理的な相互接続の結果として生じる か、あるいは特に微分代数方程式 (DAE) システムをモデル化しようとしている ために生じることがあります。

モデルに代数ループが含まれる場合、Simulink は各時間ステップごとにループ解法ルーチンを呼び出します。ループソルバは繰り返し実行して(可能であれば) 問題に対する解を求めます。その結果、代数ループをもつモデルは、それらをもたないモデルよりも実行が遅くなります。

F(z) = 0を解くために、Simulink ループソルバは、弱ライン検索および偏導関数 のヤコビアン行列に対するランク1の更新を伴うニュートン法を使用します。こ の手法はロバストですが、代数状態zに適切な初期推定を与えなければ、ループ ソルバが収束しないようなループを作成する可能性があります。代数ループ内の ラインに対する初期推定値は、その1つのライン上に(信号に対する初期条件を 指定するために通常使用する)IC ブロックを配置して指定することができます。 上に示したように、代数ループ内のラインに対する初期推定値を指定するもう1 つの方法は、Algebraic Constraint ブロックを使用することです。

ループ内の代数状態変数に対する初期推定値を指定するためには、可能であれば IC ブロックまたは Algebraic Constraint ブロックを使用してください。

非代数直接フィードスルーループ

直接フィードスルーブロックを伴う全てのループが代数的であることは、一般的 規則ですが、これに従わない例外も存在します。代数ループとならない例外と は、つぎの2点です。

- トリガ付きサブシステムを含むループ
- 出力から積分器のリセット端子へのループ

トリガ付きサブシステムは、トリガイベント間で出力定数を保持します(8-8 ページの"Triggered サブシステム"を参照)。そのため、ソルバは前の時間ス テップからの出力を使って、現在の時刻での入力を計算できます。実際に、これ はトリガ付きサブシステムを含むループに対してソルバが行うことで、そのため 代数ソルバの必要性はなくなります。

注意 ソルバはトリガ付きサブシステムの前の出力を使ってフィードバック入力 を計算するため、サブシステムとフィードバックパスのブロックは、出力では1 サンプル時間遅延します。トリガ付きフィードバックループをもつシステムのシ ミュレーション時に、Simulink はワーニングを表示してそのような遅延が生じる ことを知らせます。

たとえば、つぎのシステムを考えます。



このシステムは、つぎの方程式を効率的に解くことができます。

z=1+u

ここで、uはサブシステムにトリガが適用された最新時刻のzの値です。このシ ステムの出力は、システムのスコープで表示されるように階段関数になります。



ここで、前の例に示されるシステムからトリガを除去することを考えます。



このような場合、加算器サブシステムの u2 端子での入力は、全ての時間ステップに対して、カレントの時間ステップでのサブシステム出力と等しくなります。 このシステムの数学的な表現

z = z + 1

は、数学的には正しい解が存在しないことを示します。

離散システムのモデル化とシミュレーション

Simulink には、離散(サンプリングしたデータ)システムをシミュレーションす る機能があります。モデルは、 マルチレート にすることができます。すなわち、 異なるレートでサンプリングされたブロックを含むことができます。モデルは、 離散ブロックと連続ブロックが混在するノノイブリッドにすることができます。

離散ブロック

各離散ブロックは、その入力において組み込みサンプラをもち、出力においてゼ ロ次ホールドをもちます。離散ブロックと連続ブロックが混在するときは、サン プル時間の間における離散ブロックの出力は、一定に保たれます。離散ブロック の出力は、サンプルヒットに対応する時間でのみ更新されます。

サンプル時間

Sample time パラメータは、離散ブロックの状態が更新されるサンプル時間を設定します。通常、サンプル時間はスカラ変数に設定されます。しかし、このフィールドに2要素ベクトルを指定することによって、オフセット時間を指定することができます。

たとえば、Sample time パラメータをベクトル [Ts,offset] として指定すると、サン プル時間は Ts に設定され、オフセット値は offset に設定されます。離散ブロッ クは、サンプル時間の整数倍とオフセット値のみで更新されます。

t = n * Ts + offset

ここで、nは整数、offsetは正でも負でも構いませんがサンプル時間未満です。 オフセットは、ある離散ブロックを他のブロックより早くまたは遅れて更新させ なければならない場合に有効です。

シミュレーションの実行中に、ブロックのサンプル時間を変更することはできま せん。ブロックのサンプル時間を変更したい場合は、変更が有効になるように、 シミュレーションを一度停止して再起動する必要があります。

純粋な離散システム

純粋な離散システムは、任意のソルバを使ってシミュレーションすることができ ます。解に違いはありません。サンプルヒットのみでの出力ポイントを生成する には、discrete ソルバの1つを選択してください。

マルチレートシステム

マルチレートシステムは、異なるレートでサンプリングされるブロックを含みま す。これらのシステムは離散ブロック、あるいは離散と連続の両方のプロックを 用いてモデル化することができます。たとえば、つぎの簡単なマルチレート離散 モデルを考えます。



この例で、DTF1 Discrete Transfer Fcn ブロックの **Sample time** は、[1 0.1] に設定され、これにより 0.1 のオフセットが与えられます。DTF2 Discrete Transfer Fcn ブロックの **Sample time** は 0.7 に設定され、オフセットはありません。

シミュレーションを開始し (5-3 ページの "Running a Simulation from the Command Line" を参照)、 関数 stairs を使って出力をプロットします。

[t,x,y] = sim('multirate', 3); stairs(t,y)

この結果、つぎのプロットが選られます。



オフセットが 0.1 の DTF1 ブロックに対して、t = 0.1 になるまで出力はありません。伝達関数の初期条件はゼロなので、DTF1 の出力 y(1) はこの時間まではゼロです。

離散システムのステップサイズの決定

離散システムのシミュレーションでは、シミュレータが*サンプル時間ヒット、*す なわち、もっとも短いサンプル時間の整数倍ごとにシミュレーションのステップ をとることが必要です。そうでない場合は、シミュレータはシステムの状態にお いて重要な遷移を失うことがあります。Simulink はシミュレーションステップサ イズを選択することによって、サンプル時間ヒットを伴うステップを保証し、こ のことを防ぎます。Simulink が選択するステップサイズは、システムの基本サン プル時間とシステムのシミュレーションに用いるソルバのタイプにより異なりま す。

離散システムの*基本的サンプル時間*は、システムの実際のサンプル時間の最大 公約数です。たとえば、システムのサンプル時間が 0.25 秒と 0.5 秒であるとしま す。この場合の基本サンプル時間は、0.25 秒です。また、サンプル時間が 0.5 秒 と 0.75 秒とします。この場合、基本サンプル時間は、0.25 秒です。

離散システムを解くために、固定ステップ離散ソルバまたは可変ステップ離散ソ ルバのいずれかを利用することを指定することができます。固定ステップソルバ は、シミュレーションステップサイズを離散システムの基本サンプル時間と等し く設定します。可変ステップソルバは、実際のサンプル時間ヒット間の距離に等 しくなるようにステップサイズを変化させます(可変ステップサイズは、常に基 本ステップサイズの整数倍です)。つぎの図は、固定ステップソルバと可変ス テップソルバの違いを示しています。



Variable-Step Solver

図中で、矢印はシミュレーションステップを示し、円はサンプル時間ヒットを表わします。図が示すように、可変ステップソルバは、基本サンプル時間がシミュレートするシステムの実際のサンプル時間よりも短い場合は、小さいシミュレーションステップが必要です。一方、固定ステップソルバは、システムのサンプル時間が基本サンプル時間である場合は、実現のために要するメモリ量が少なく、

高速です。これは、(Real-Time Workshop を使った)Simulink モデルからのコード 生成を伴うアプリケーションにおいては、利点となります。

サンプル時間の伝播

つぎの図は、Gain ブロックに接続しているサンプル時間が Ts の Discrete Filter ブロックを示しています。



Gain ブロックの出力は、入力の定数倍なので、出力はフィルタと同じレートで 変化します。言い換えると、Gain ブロックは、フィルタのサンプルレートと等 しい有効なサンプルレートをもちます。これは、Simulink におけるサンプル時間 の伝播での基本的なメカニズムです。

Simulink は、つぎの規則に従って、個々のブロックにサンプル時間を設定します。

- Continuous ブロック(例、Integrator, Derivative, Transfer Fcn 等)は、定義により連続です。
- Constant ブロック (たとえば、Constant) は、定義により定数です。
- Discrete ブロック(例、Zero-Order Hold, Unit Delay, Discrete Transfer Fcn 等)は、ブロックのダイアログボックスで明示的に指定されたサンプル時間をもちます。
- それ以外のすべてのブロックは、入力のサンプル時間に基づく暗示的に定義 されたサンプル時間をもちます。たとえば、Integrator ブロックの後の Gain ブ ロックは、連続ブロックとして扱われ、Zero-Order Hold の後の Gain ブロック は Zero-Order Hold ブロックと同じサンプル時間をもつ離散ブロックとして扱 われます。

入力が異なるサンプル時間をもつブロックに対して、すべてのサンプル時間 が最も速いサンプル時間の整数倍である場合、ブロックは最も速い入力のサ ンプル時間に割り当てられます。可変ステップソルバが用いられている場合 は、ブロックは連続サンプル時間を割り当てられます。固定ステップソルバ が用いられていて、サンプル時間の最大公約数(基本サンプル時間)が計算可 能な場合は、その値が用いられます。そうでない場合は、連続サンプル時間 が用いられます。

ある状況下では、Simulink は source ブロックへのサンプル時間の逆伝播がシ ミュレーションの出力に影響を与えずに行うことができる場合は、行います。 たとえば下記のモデルで、Simulink は、Signal Generator ブロックが Discrete-Time Integrator ブロックに接続しているので、Signal Generator ブロックと Gain ブロッ クに Discrete-Time Integrator ブロックと同じサンプル時間を割り当てることを認 識します。



このことは、Simulink の Format メニューから Sample Time Colors を可能にして、すべてのブロックを赤に色付けすることによって確認することができます。 Discrete-Time Integrator ブロックはサンプル時間で入力を見るだけなので、この 変更はシミュレーションの結果に影響を与えませんが、性能は改善されます。

以下に示すように Discrete-Time Integrator ブロックを連続の Integrator ブロックと 置換え、Edit メニューから Update diagram を選択してモデルを再度色付けする と、Signal Generator と Gain ブロックが黒に色づけされ、連続ブロックに変わっ ていることが分かります。



不変定数

ブロックは明確に定義されたサンプル時間をもつか、または接続されているブロックか、または接続するブロックからサンプル時間を継承します。

Simulink は、Constant ブロックに無限大のサンプル時間を割り当てています。そのため、それを定数サンプル時間と呼んでいます。他のブロックが、Constant ブロックからの出力を入力とし、その他のブロックのサンプル時間を継承しない場合、定数サンプル時間をもちます。このことは、モデルユーザによってパラメータが明示的に変更されない限り、このようなブロックの出力は、シミュレーション中、変化しないことを意味しています。

たとえば、つぎのようなモデルでは、Constant ブロックと Gain ブロックが定数 サンプル時間をもつことになります。



Simulink はシミュレーション中に、ブロックパラメータを変更できる機能をもっているので、定数サンプル時間をもつブロックも含め全てのブロックについて、 そのモデルにとって有効なサンプル時間で出力を生成できます。

注意 Format メニューから Sample Time Colors を選択すると、定数サンプル時間をもつブロックが見つけられます。定数サンプル時間を持つブロックは、マジェンタ色になります。

この機能を使って、全てのブロックがそれぞれのサンプル時間時に、出力を計 算するか、または純粋に連続的なシステムの場合では、全てのシミュレーション ステップで出力を計算します。シミュレーション中にパラメータが変化しない定 数サンプル時間をもつブロックに対して、シミュレーション中にこれらのブロッ クをステップ毎に計算することは非効率で、シミュレーション速度の低下につな がってしまいます。

Simulink のインラインパラメータオプション (5-29 ページの"インラインパラ メータ"を参照)を設定すると、シミュレーションの"ループ"から、定数サン プル時間をもつブロックすべての取り除くことができます。この機能は、つぎの 2 つの効果をもたらします。第1に、このようなブロックに対するパラメータ は、シミュレーション中に変更されなくなります。第2に、シミュレーション速 度が改善されます。シミュレーション速度の改善は、モデルの複雑さ、定数サン プル時間をもつブロック数、シミュレーションの有効サンプリングレートに依存 します。

連続および離散混在システム

連続および離散混在システムは、サンプリングされるブロックと連続ブロックから構成されます。このようなシステムは、任意の積分手法を使って、シミュレーションすることができますが、積分手法によって効率や精度が異なります。ほとんどの連続および離散混在システムでは、Runge-Kuttaの可変ステップ手法のode23とode45が、効率および精度において他の手法よりも優れています。離散 プロックのサンプルアンドホールドに関連する不連続性のため、ode15sとode113の手法は、連続および離散混在システムに対して推奨できません。
4

モデルの作成

Simulink の起動		•		•		•	•	•	•	•	•	•	•	•	•	•	•		. 4-2
オブジェクトの選	劉沢 .	•				•					•		•	•		•	•	•	. 4-8
プロック		•		•		•					•		•	•		•			.4-10
ブロックの接続		•		•									•						.4-24
信号の利用		•		•		•	•		•	•		•	•		•				.4-30
注釈		•		•		•	•		•	•		•	•		•				.4-44
データタイプの様	钱能 .	•		•	•								•						.4-45
データオブジェク	っトの	機能	19 19	•		•	•		•	•		•	•		•				.4-51
マウスとキーボー	-ドの	動化	ፑወ	ま	28	め	•		•	•		•	•		•				.4-63
サブシステムの作	₣成.	•		•	•	•													.4-66
コールバックルー	・チン	の何	吏用	法	•								•						.4-71
モデル作成のヒン	/ ト.	•				•													.4-76
ライプラリ		•		•		•	•		•	•		•	•		•				.4-77
方程式のモデル伯	Ľ.	•		•		•	•		•	•		•	•		•				.4-87
モデルの保存		•				•													.4-90
ブロック線図の日	「刷.	•				•													.4-91
モデルの検索とこ	ブラウ	ズ		•		•	•		•	•	•	•	•	•	•	•	•	•	.4-95
モデルのバージョ	りと管	理		•		•							•						4-106
Simulink セッショ	ョンの)終「	了.																4-116

Simulink の起動

Simulink を起動するには、まず MATLAB を起動する必要があります。より多く の情報は、MATLAB のドキュメントを参照してください。さて、Simulink をつ ぎの2つの方法で起動します。

- MATLAB ツールバーの Simulink アイコン 上手をクリック
- MATLAB プロンプトでコマンド simulink を入力

Microsoft Windows のプラットホーム上では、Simulink を起動すると Simulink ラ イブラリブラウザが現れます。

👿 Simulink Library Browser		_ 🗆 🗡
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>H</u> elp		
🗋 🚔 — 🎘 Find		
Continuous: simulink3/Continuous		
	\sim	Continuous
Discrete	+	
		Discrete
Math	11- f († 11)	Functions & Tables
Signals & Systems	+	
	17	Math
Stateflow	T	Nonlinear
	<u>, 1</u>	o: 1.4.0.1
	ir Ust	Signals & Systems
	×↓≠ →•≠	Sinks
	+ <u>**</u> *	
		Sources
Beadu	+	
rioddy		

ライブラリブラウザは、システムにインストールされている Simulink ブロック ライブラリをツリー構造で表示します。ライブラリブラウザからモデルウィンド ウにブロックをコピーすることによって、モデルを作成します(この手順の詳細 については、本章で後述します)。 UNIX のプラットホーム上では、Simulink を起動すると Simulink ブロックライブ ラリウィンドウが現れます。



Simulink ライブラリウィンドウは、Simulink 自身も含んだブロックライブラリを 表すアイコンを表示します。モデルウィンドウにライブラリからブロックをコ ピーすることでモデルを作成することができます。

注意 Windows では、ライブラリブラウザウィンドウで Simulink ノードを右ク リックすることで Simulink ライブラリウィンドウを表示することができます。

新規モデルの作成

新規モデルを作成するためには、ライブラリブラウザ (Windows のみ) のツール バーの New ボタンをクリックするか、ライブラリウィンドウの File メニューか ら New を選択し、Model を選びます。他のウィンドウと同じようにウィンドウ は移動させることができます。第2章の"クイックスタート"では、簡単なモデル を作成する方法について説明しています。4-77 ページの"ライブラリ"では、方 程式をモデル化するシステムを作成する方法について説明します。

既存のモデルの編集

既存のモデル線図は、つぎのいずれかの方法で編集します。

- ライブラリブラウザ(Windowsのみ)のツールバーの Open ボタンを選択するか、 Simulink ライブラリウィンドウの File メニューから Open コマンドを選択し、 編集したいモデルのモデルファイル名を選択するか入力します。
- MATLAB コマンドウィンドウに(.mdl 拡張子をつけないで)モデルの名前を入 力します。モデルはカレントディレクトリまたはパス上になければなりません。

Simulink コマンドの実行

コマンドを実行することで、Simulink を起動しモデルを動作させることができま す。つぎのようにして、コマンドを実行することができます。

- Simulink メニューバーから項目を選択
- Simulink コンテキストメニューから項目を選択 (Microsoft Windows のみ)
- Simulink ツールバー上のボタンをクリック (Windows のみ)
- MATLAB コマンドウィンドウでコマンドを入力

コマンド実行のための Simulink メニューバーの利用

Simulink メニューバーは、各モデルウィンドウの上側に表示されます。メニューコマンドは、そのウィンドウの内部に対して適用されます。

コマンド実行のためのコンテキストメニューの利用

Windows版の Simulink では、モデルウィンドウやブロックライブラリウィンド ウ上で右マウスボタンをクリックすると、コンテキストメニューが表示されま す。メニューの内容はブロックが選択されているかどうかに依存して変化しま す。ブロックが選択されていない場合、モデルやライブラリ全体に対して適用さ れるコマンドがメニュー表示されます。

コマンド実行のための Simulink ツールバーの利用

Windows 版の Simulink では、モデルウィンドウの Simulink メニューバーの下側 にツールバーがオプション表示されます。ツールバーを表示するためには、 Simulink View メニューの Toolbar オプションをチェックします。

🔁 untitled		_ 🗆 ×		
<u>File Edit View Simulation F</u>	orma <u>t</u> T <u>o</u> ols			
D 🛎 🖬 🎒 X 🖻	e ⊇ e >	=		— Toolbar

ツールバーには、よく利用される Simulink コマンドに関連したボタンが含まれています。たとえば、モデルのオープンやシミュレーションの実行、モデルのク

ローズなどです。関連するボタンをクリックすることでそのようなコマンドを実 行することができます。たとえば、Simulink モデルをオープンするためには、 オープンフォルダアイコンを含むボタンをクリックします。ボタン上にマウスポ インタを合わせることで、そのボタンがどのようなコマンドを実行するかを確認 することができます。そのボタンを説明するテキストを含む小さなウィンドウが 現れます。このウィンドウをツールチップと呼びます。ツールチップは、ツール バー上の各ボタン上でマウスポインタを、しばらく静止していると表示されま す。Simulink View メニューで Toolbar オプションを非選択にすることで、ツー ルバーを非表示にすることができます。

コマンドを実行するための MATLAB ウィンドウの利用

シミュレーションを実行し結果を解析するために、MATLAB コマンドウィンド ウで MATLAB コマンドを実行することができます。シミュレーションの実行に 関しては 第5章で議論され,シミュレーション結果の解析に関しては 第6章の" シミュレーション結果の解析"で議論されます。

コマンドを元に戻す

Edit メニューから Undo を選択することで、連続した 101 回までの操作の結果を キャンセルすることができます。以下の操作を元に戻すことが可能です。

- ブロックの追加または削除
- ラインの追加または削除
- モデルの注釈の追加または削除
- ブロック名の編集
- サブシステムの作成

Edit メニューから Redo を選択することで、Undo コマンドの逆の操作を行うことができます。

Simulink ウィンドウ

Simulink は、ブロックライブラリブラウザ、ブロックライブラリ、モデル、グラフィカルなシミュレーション出力(スコープ)を、別々のウィンドウを使って表示します。これらのウィンドウは、MATLABのFigure ウィンドウとは異なり、Handle Graphics[®] コマンドを使用して操作することはできません。

Simulink ウィンドウは、最も一般的なスクリーン解像度に対して適切な大きさに なっています。非常に高い/低い解像度のモニタで利用する場合、ウィンドウサ イズが小さ過ぎたり、大き過ぎたりするかもしれません。そのような場合、ウィ ンドウの大きさを変更し、新しいウィンドウサイズを保存するためにモデルを保 存してください。

ステータスバー

Windows バージョンの Simulink は、各モデルとライブラリウィンドウの下部に ステータスバーを表示します。



シミュレーションの実行中、ステータスバーには現在のシミュレーション時間や ソルバの名前など、シミュレーションの状態が表示されます。Simulink View メ ニューの Status Bar を選択 / 非選択することで、ステータスバーの表示 / 非表示 を決定することができます。

プロック線図のズーミング

Simulink では、現在の Simulink ウィンドウのブロック線図の表示の拡大と縮小ができます。表示をズームするためには、つぎのようにします。

- 表示を拡大するためにViewメニューからZoomInを選択します(あるいは、rを タイプします。
- 表示を縮小するために View メニューから Zoom Out を選択します(あるいは、v をタイプします。
- Select 表示を適切な大きさにするために View メニューから Fit System to View(ウィンドウサイズにあわせて表示)を選択します(あるいはスペースバーを押 します)。
- 実際の大きさでダイアグラムを表示するためにViewメニューからNormal (標準 100%)を選択します。

モデルブラウザのコンテンツペインか分割ウィンドウでダイアグラムをオープン するとき、Simulink はデフォルトでブロック線図を適切な大きさで表示します。 ダイアグラムのズームの設定を変更すると、Simulink はダイアグラムを閉じると きに設定を保存し、つぎにダイアグラムをオープンするときに設定を復元しま す。デフォルトの設定に戻したい場合、つぎにダイアグラムをオープンするとき に View メニューから Fit System to View(ウィンドウサイズにあわせて表示)を 選択します。

オブジェクトの選択

ブロックのコピーやラインの削除など、モデル作成動作の多くでは、まず1つまたは複数のブロックやライン(オブジェクト)を選択する必要があります。

1つのオブジェクトの選択

1 つのオブジェクトを選択するには、それをクリックします。1 つのオブジェクトを選択するには、それをクリックします。選択したブロックの4隅と選択したラインの両端近くに、小さな黒い四角の"ハンドル"が表示されます。たとえば、下の図は、選択した Sine Wave ブロックと選択したラインを示しています。



オブジェクトをクリックして選択すると、その他の選択オブジェクトは選択解除 されます。

複数のオブジェクトの選択

オブジェクトを1つずつ選択するか、境界ボックスを用いて互いに隣接したオ ブジェクトを選択するか、あるいはモデル全体を選択することによって、複数の オブジェクトを選択することができます。

1つずつ複数のオブジェクトを選択する

各オブジェクトを個々に選択することによって複数のオブジェクトを選択するには、Shift キーを押したまま選択する各オブジェクトをクリックします。選択されているオブジェクトを選択解除するには、Shift キーを押しながらオブジェクトを再度クリックします。

境界ボックスを用いて複数のオブジェクトを選択する

ウィンドウの同じ領域内にある複数のオブジェクトを選択する簡単な方法は、オ ブジェクトのまわりに境界ボックスを描くことです。 1 境界ボックスの一隅にポインタを位置付けすることによって境界ボックスを 開始する位置を決め、つぎにマウスボタンを押したままにします。カーソル の形に注意してください。



2 ポインタを境界ボックスの反対側の隅にドラッグします。選択されるブロックとラインが、点線の長方形で囲まれます。

Sine Wave	Scope

3 マウスボタンを解除します。少なくとも部分的に境界ボックスで囲まれたす べてのブロックとラインが選択されます。



モデル全体の選択

アクティブウィンドウ内のすべてのオブジェクトを選択するには、Edit メ ニューから Select All を選択します。この方法でプロックやラインを選択するこ とによってサブシステムを作成することはできません。詳細については、4-66 ページの"サプシステムの作成"を参照してください。

ブロック

ブロックは、Simulink モデルを構築するための要素です。適切な方法でブロック を作成し結線することで、実質的にどんな動的システムでもモデル化することが できます。この節では、動的システムをモデル化するためにどのようにブロック を利用するのかを説明します。

ブロックのデータ情報

Microsoft Windows では、Simulink は、ブロック線図に表示されているブロック 上にポインタを合わせることで、ブロックに関する情報をポップアップウィンド ウに表示します。この機能を無効にしたり、どのようなブロック情報を表示する かをコントロールするために、Simulink View メニューの Block Data Tips(プロッ クデータティップスオプション)を選択します。

バーチャルブロック

モデルを作成するとき、Simulink ブロックが2つの基本的なカテゴリに分類され ることを知っておく必要があります。それは、ノンバーチャルとバーチャルで す。ノンバーチャルブロックは、システムのシミュレーションにおいて動的な役 割を果たします。ノンバーチャルブロックを加えたり削除したりすると、モデル の振る舞いが変化します。バーチャルブロックはその逆で、シミュレーションに おいては動的な役割は果たしません。単に、視覚的にモデルを構成するのに役立 ちます。いくつかのブロックは、ある状況下ではバーチャルで、その他の状況で はノンバーチャルになります。このようなブロックを、条件付きバーチャルブ ロックと呼びます。つぎの表は、Simulinkのバーチャルプロックと条件付きバー チャルブロックをリストします。

ブロック名	ブロックがバーチャルに動作するための条件
Bus Selector	常にバーチャル
Data Store Memory	常にバーチャル
Demux	常にバーチャル
Enable Port	常にバーチャル
From	常にバーチャル

表 4-1: バーチャルプロックと条件付きバーチャルプロック

プロック名	ブロックがバーチャルに動作するための条件
Goto	常にバーチャル
Goto Tag Visibility	常にバーチャル
Ground	常にバーチャル
Inport	条件付き実行サブシステムの中に存在し、 <i>かつ</i> 、 Outport ブロックに直接結線されている場合 <i>以外は</i> 常 にバーチャル
Mux	常にバーチャル
Outport	サブシステムブロック (条件付きかそうでないかに 関わらず)の中に存在し、かつ、Simulink ウィンドウ のルート (トップレベル)に存在 <i>しない</i> 場合、バー チャル
Selector	行列モードの場合以外はバーチャル
Subsystem	ブロックが条件付きで実行される時と、ブロックの Treat as Atomic Unit オプションが選択されている時 以外は、バーチャル
Terminator	常にバーチャル
Test Point	常にバーチャル
Trigger Port	出力端子が表示されて <i>いない</i> 場合、バーチャル

表 4-1: バーチャルブロックと条件付きバーチャルブロック (Continued)

ウィンドウ間でのブロックのコピーと移動

モデルを作成する際に、Simulink ブロックライブラリまたは他のモデルウィンド ウからモデルウィンドウにブロックをコピーすることが多くあります。これはつ ぎの手順で行います。

- 1 該当するブロックライブラリまたはソースモデルウィンドウを開きます。
- 2 コピーしたいブロックをコピー先のモデルウィンドウまでドラッグします。
 ブロックをドラッグするためには、ブロックアイコンにカーソルを合わせ、

マウスボタンを押したままにします。カーソルをコピー先のウィンドウに移動し、マウスボタンを解除します。

Simulink Library Browser からモデルウィンドウにブロックをドラッグすることも できます。詳細は4-83ページの"ブロックライブラリの参照"をご参照ください。

注意 Simulink は、sum,mux,demux,bus selector ブロックを Simulink ブロックライ プラリからモデルにコピーした時、その名前を非表示にします。これは、モデル ダイアグラムの不必要な煩雑さを避けるためです(これらのブロックの形は、そ れぞれの機能を明示的に示しています)。

Edit メニューから Copy/Paste コマンドを利用してブロックをコピーすることも できます。

- 1 コピーしたいブロックを選択します
- 2 Edit メニューから Copy を選択します
- 3 ターゲットのモデルウィンドウをアクティブにします
- 4 Edit メニューの Paste を選択します

Simulink は、コピーしたそれぞれのブロックに名前を割り当てます。モデル内で そのタイプのブロックがそれだけである場合には、ソースウィンドウでの名前と 同じ名前が付けられます。たとえば、Linear ライブラリから Gain ブロックをモ デルウィンドウにコピーする場合、新しいブロックの名前は Gain となります。 モデルの中に、すでに Gain という名前のブロックが存在している場合には、 Simulink はブロック名に連続番号を追加します(たとえば、Gain1、Gain2)。ブ ロックの名前は変更できます。4-18 ページの"ブロック名の操作".を参照してく ださい。

ブロックのコピーの際には、新規ブロックはオリジナルのブロックのすべてのパ ラメータ値を引き継ぎます。

Simulink は、目に見えない5ピクセルグリッドを用いて、ブロックの整列を簡単にします。モデル内部のすべてのブロックは、グリッド上のラインにぴったりと合わせられます。ブロックを選択し、矢印キーを押すことによって、ブロックを上下左右にわずかに移動することができます。

MATLAB ウィンドウ内で以下のコマンドをタイプインすることにより、モデル ウィンドウにグリッドを表示することができます。

set_param('<model name>','showgrid','on')

グリッド間隔を変更するためには、つぎのようにタイプインします。

set_param('<model name>','gridspacing',<number of pixels>)

たとえば、グリッド間隔を 20 ピクセルに変更するためは、つぎのようにタイプ インします。

set_param('<model name>','gridspacing',20)

上記の2つのコマンドで、<model name>をタイプインする代わりに、モデルを 選択し、gcs とタイプインすることもできます。

Copy、Cut、Paste コマンドを用いて、(ワープロプログラムなど)互換性のある アプリケーションにブロックをコピーしたり移動したりすることができます。こ れらのコマンドは、ブロックのグラフィックス表現のみをコピーし、そのパラ メータはコピーしません。

ウィンドウ間でのブロックの移動は、ブロックを選択する際に Shift キーを押す ことを除いて、ブロックのコピーと同様です。

Edit メニューから Undo コマンドを使って、追加したブロックを削除することが できます。

モデル内でのブロックの移動

モデルウィンドウ内で1つのブロックをある場所から別の場所に移動するには、 ブロックを新しい位置までドラッグします。Simulinkは、移動したブロックに接 続されているラインを自動的に再配置します。

接続線を含め、複数のブロックを移動するには、つぎのようにします。

- ブロックとラインを選択します。複数のブロックを選択する方法については、 4-8ページの"複数のオブジェクトの選択"を参照してください。
- 2 オブジェクトを新しい位置にドラッグし、マウスボタンを解除します。

モデル内でのブロックの複製

つぎの手順に従うことにより、モデル内のブロックを複製することができます。 Ctrl キーを押しながら左マウスボタンでブロックを選択し、それを新しい位置 までドラッグします。これは、右マウスボタンを使用してブロックをドラッグす ることによっても行えます。複製したブロックは、オリジナルのブロックと同じ パラメータ値をもっています。新しいブロック名には連続番号が付けられます。

ブロックパラメータの指定

Simulink ユーザインタフェースで、ブロックのパラメータに値を割り当てること ができます。(A-7 ページの"共通ブロックのパラメータ"を参照してください)。 パラメータを設定するために、4-15 ページの"ブロックプロパティダイアログ ボックス"を利用します。また、多くのブロックは、設定可能な1つ以上のブ ロック特有なパラメータをもっています(A-10 ページの"ブロック固有のパラ メータ"を参照してください)。これらのパラメータを設定することで、モデル の必要条件を満たすようなブロックの動きをカスタマイズすることができます。

ブロック固有のパラメータの設定

ブロック特有のパラメータをもつブロックは、すべて、パラメータを参照したり 設定するために使用するダイアログボックスをもっています。このダイアロブ ボックスは、モデルウィンドウ内のブロックを選択することによって、または、 モデルウィンドウの Edit メニューからか、モデルウィンドウのコンテキストメ ニュー(右クリック)から BLOCK Parameters を選択することで表示できます。 ここでの BLOCK とは、選択したブロックの名前、たとえば、Constat Parameters のことです。また、ブロックパラメータダイアログボックスは、モ デル内や Library ウィンドウ内のアイコンをダブルクリックしても表示すること ができます。

注意 これは、サブシステムブロックを除いたパラメータダイアログボックスを もつすべてのブロックに当てはまります。サブシステムのブロックパラメータダ イアロブを表示するためには、モデルウィンドウの Edit メニューかコンテキス トメニューを使用してください。

特定のブロックのパラメータダイアログボックスの情報は、第9章の"ブロック リファレンス"のブロックの説明を参照してください。

どんなブロックパラメータも、Simulinkのset_paramコマンドを使って設定する ことができます。詳細は、10-27ページのset_paramを参照してください。

ブロックパラメータダイアログのパラメータの値や、set_param コマンドの値を 指定するときに、適当な結果かどうかを評価する任意の MATLAB の定数、変 数、パラメータとして Simulink のデータオブジェクトを評価するような変数や 表現も使用できます (4-54 ページの " パラメータとしてデータオブジェクトを使 用 "を参照してください)。

ブロックプロパティダイアログボックス

すべてのブロックで、ブロックのプロパティ、つまり、パラメータを設定するために、このダイアログボックスを使用します。このダイアログボックスを表示するには、モデルウィンドウ内のブロックを選択し、つぎに、モデルウィンドウの Edit メニューを選択してください。Edit メニューには、BLOCK Parameters という項目が含まれます。ここで、BLOCK は、選択したブロックの名前で、たとえば、Constat Properties です。ブロックのプロパティダイアログボックスを表示するために、この項目を選択します。

いくつかの共通のパラメータを設定するために、Block Properties ダイアログボックスを利用します。

Block Properties: Integrator					
Info					
Toescription' is a text field associated with the block that is generally used for saving comments about about the block usage within the model "Priority' can be empty or an integer value which specifies the block's sequencing during execution relative to other blocks with priorities in the same window. "Tag' is a general text field which is saved with the block "Open function" is the function to be called when you double-click on a block. "Attributes format string' sets the display format shown below the block name (e.g., Pri=% <priority>).</priority>					
Properties					
Description:					
p. Prioritur					
r nony.					
Lac					
Open function:					
Attributes format string:					
OK Cancel <u>H</u> elp <u>Apply</u>					

このダイアログは、つぎのフィールドをもっています。

Description(概要)

ブロックの目的の概略を記述します。

Priority(優先順位)

モデルにおけるこのブロックの実行プライオリティを他のブロックとの関連で指 定します。詳細は、4-20ページの"ブロックプライオリティの割り当て"をご参照 ください。

Tag(タグ)

ブロックと一緒に保存される一般的なテキストフィールド。

Open function(オープン関数)

このブロックをオープンするときにコールされる MATLAB 関数を指定します。

Attributes format string(属性フォーマット文字列)

ブロックの AttributesFormatString パラメータを設定します。このパラメータは、 ブロックアイコンの下に表示するパラメータを指定します。付録 A では、ブロッ クがもつパラメータについて説明します。AttributesFormatString パラメータを 使ってブロックのアイコンの下にある指定したパラメータの値を表示することが できます。

属性フォーマット文字は、組み込みパラメータ名のテキスト文字です。組み込み パラメータ名は、%<>で囲まれたパラメータ名になります。たとえば、 %<priority>です。Simulink は、プロックアイコンの下に属性フォーマット文字 を表示し、各パラメータ名を関連するパラメータ値で置き換えます。改行して各 パラメータを表示するために、ラインフィードキャラクタ(\n)を利用することが できます。たとえば、属性フォーマット文字

pri=%<priority>\ngain=%<Gain>

を指定すると、Gainブロックの表示は、つぎのようになります。

1-> Gain pri=10 gain=1

パラメータ値が文字、または、数値でない場合、Simulink はパラメータ値の部分 に N/S(未サポート)を表示します。パラメータ名が無効な場合、Simulink は "???"を表示します。

ブロックの削除

1 つまたは複数のブロックを削除するには、削除するブロックを選択し、Delete キーまたは Backspace キーを押します。また、Edit メニューから Clear または Cut を選択することもできます。Cut コマンドはブロックをクリップボードに書き込 むので、それらをモデルに貼り付けることができます。Delete キーや Backspace キー、または Clear コマンドを用いた場合には、後でブロックを貼り付けること はできません。

Edit メニューから Undo コマンドを使って、削除したブロックを元の位置に戻す ことができます。

ブロックの方向の変更

デフォルトでは、信号はブロックの左から右へと流れていきます。入力端子は左 側にあり、出力端子は右側にあります。ブロックの方向は、Format メニューか らつぎのコマンドの一つを選択することによって変更することができます。

- Flip Block コマンドは、ブロックを 180 度回転させます。
- Rotate Block コマンドは、ブロックを時計周りに 90 度回転させます。

下の図は、Rotate Block および Flip Block メニュー項目を用いてブロックの方向 を変更した後、Simulink がどのように端子を並べるかを示したものです。ブロッ クの中のテキストが方向を示しています。



ブロックのサイズ変更

ブロックのサイズを変更するには、ブロックを選択し、選択状態を示すハンドル のどれか1つをドラッグします。マウスボタンを押している間、点線の長方形で 新しいブロックサイズが示されます。マウスボタンを解除すると、ブロックのサ イズが変更されます。

たとえば、下の図にサイズを変更している Signal Generator ブロックを示します。 右下のハンドルが選択され、カーソル位置までドラッグされています。マウスボ タンを解除すると、ブロックは新しいサイズになります。この図はサイズを変更 しているブロックを示しています。



ブロック名の操作

モデル内のすべてのブロック名は一意的であり、少なくとも1つのキャラクタ を含んでいなければなりません。デフォルトでブロック名は、つぎの図に示すよ うに、両側に端子があるブロックの下か、あるいは上下に端子があるブロックの 左側に表示されます。



ブロック名の変更

ブロック名は、つぎのいずれかの方法で編集することができます。

- Microsoft Windows、または、UNIX システムでブロック名を変更するためには、 ブロック名をクリックし、その後ブロック名を選択するためにダブルクリッ クするか、カーソルをドラッグします。最後に新しい名前を入力します。
- キャラクタを挿入するには、2つのキャラクタの間をクリックして挿入位置を 決め、テキストを挿入します。
- 文字を置き換えるには、マウスをドラッグして置き換えるテキストの範囲を 選択し、新しいテキストを入力します。

モデル内のどこか別の場所でポインタをクリックするか、別の動作をすると、名 前は確定されるか拒否されます。すでに存在している名前やキャラクタを含まな い名前にブロック名を変更しようとすると、Simulink はエラーメッセージを表示 します。

ブロック名に使用しているフォントを変更するには、ブロックを選択し、Format メニューから Font メニュー項目を選択します。Set Font ダイアログボックスから フォントを選択します。この手順では、ブロックアイコン上のテキストのフォン トも変更されます。

Edit メニューから Undo を選択して、ブロック名の編集をキャンセルすることが できます。

注意 ライブラリブロックの名前を変更すると、そのブロックに対するすべての リンクが無効になります。

ブロック名の位置の変更

選択したブロックの名前の位置は、つぎのいずれかの方法で変更することができ ます。

- ブロック名をブロックの反対側にドラッグする。
- Format メニューから Flip Name コマンドを選択する。このコマンドは ブロック 名の位置をブロックの反対側に変更します。

ブロックの方向の詳細については、4-17ページの"ブロックの方向の変更"を参照 してください。

ブロック名の表示 / 非表示の変更

選択したブロック名を表示するかどうかを変更するには、Format メニューから メニューアイテムを選択します。

- Hide Name メニューアイテムは、表示されているブロック名を非表示にします。Hide Name は、ブロックが選択されると、Show Name に変更されます。
- Show Name メニューアイテムは、非表示のブロック名を表示します。

ブロックアイコンの下側にパラメータを表示

Simulink は、ブロック線図内で、ブロックアイコンの下側に一つ、または、複数 のブロックパラメータを表示させることができます。つぎの方法で表示させるパ ラメータを指定することができます。

- ブロックのBlock PropertiesダイアログのAttributes format stringフィールドで属性 フォーマット文字を入力します (4-15 ページの"ブロックプロパティダイアロ グボックス")をご参照ください。
- tset_param (10-27 ページの set_param を参照してください)を利用して、ブロックの AttributesFormatString プロパティの値をフォーマット文字に設定します。

ブロックの切り離し

ブロックをその接続線から切り離すには、Shift キーを押しながらブロックを選択し、新しい位置までドラッグします。

ブロックプライオリティの割り当て

モデルの中のノンバーチャルブロックに実行順序を割り当てることができます。 優先順位の高いブロックは、優先順位の低いブロックの前に実行されますが、優 先順位を割り当てていないブロックの前に実行されることは必ずしもありませ ん。

ブロックプライオリティをインタラクティブに、またはプログラム的に割り当て ることができます。プライオリティをプログラムで設定するために、つぎのコマ ンドを利用します。

set_param(b,'Priority','n')

ここで、b はブロックのパス、n は適当な整数です(負の数値と0 は有効なプラ イオリティ値です)。より小さな数値が優先順位が高くなります。つまり、2 は 3 よりも優先順位が高くなります。インタラクティブにブロックのプライオリ ティを設定するために、Block Properties ダイアログの Priority フィールドにプ ライオリティを入力します。(4-15ページの"ブロックプロパティダイアログボッ クス"を参照してください)。

Simulink は、指定したブロックが、Simulinkのブロックソートアルゴリズムと矛 盾がない場合にのみ、その優先順位を守ります(3-11 ページの"ブロックのアッ プデート順の決定"を参照してください)。指定した優先順位に矛盾があると、 Simulink は指定された優先順位を無視し、適切なブロック実行順位で実行しま す。Simulink がブロック優先順位を守らない場合は、ブロック優先順位違反の診 断メッセージが表示されます(5-25 ページの"Diagnostics ページ"を参照してくだ さい)。

ブロック実行順序の表示

シミュレーション中にブロックの実行順序を表示するには、SimulinkのFormat メニューから Execution order(実行順序)を選択します。このオプションを選択 すると、ブロックダイアグラムの各ブロックの右角に番号が表示されます。



番号は、ダイアグラム内のほかのブロックに関連させた実行順序を示していま す。たとえば、1は、すべての時間ステップで最初に実行されるブロックを示 し、2はすべての時間ステップで2番目に実行されるブロックを示しています。

ドロップシャドウの使用法

ブロックを選択し、Format メニューから Show Drop Shadow を選択して、ブロックに影を付けることができます。影の付いたプロックを選択すると、メニュー項目は Hide Drop Shadow に変わります。つぎの図に、影の付いた Subsystem プロックを示します。



サンプル時間の色分け

Simulink は、ブロックが作用するサンプルレートを示すために、モデル内のブ ロックやラインをカラーコーディングすることができます。

表 4-2: サンプル時間の色分け

色	使用
黒	連続ブロック
マゼンタ	定数ブロック
黄色	ハイブリッド(ブロックをグループ化しているサブシステム や、可変サンプリング時間をもった信号をグループ化した Mux、Demux プロック)
赤	最速の離散サンプリング時間
緑	2番目に高速な離散サンプリング時間
青	3番目に高速な離散サンプリング時間
水色	4番目に高速な離散サンプリング時間
深緑	5番目に高速な離散サンプリング時間
オレンジ	6番目に高速な離散サンプリング時間
シアン	トリガー付きサブシステム内のブロック
グレー	小さなステップで固定

サンプル時間の色分け表示をするには、Format メニューから Sample Time Colors を選択してください。

Simulink は、モデルが変更されても、自動的に色分け表示の再描画を行いません。そこで、モデルの色を変更するために、Edit メニューから Update Diagram を選択してください。オリジナルの色に戻すには、Sample Time Colors を選択し、サンプル時間の色分け表示を非選択にしてください。

サンプル時間の色分け表示をするときは、それぞれのブロックに割り当てられた 色は、モデル内の他のブロックのサンプル時間との対比で決定されます。 Mux と Demux ブロックは、グループ分けのための演算子にすぎず、これらのブ ロックを通る信号は、信号のタイミング情報を保持していることに注意してくだ さい。このため、Demux ブロックから出る線が異なるサンプリング時間をもつ ソースから出ている場合は、異なる色を持つこともありえます。この場合、Mux および Demux ブロックは、マルチレートの信号を示すハイブリッド(黄)色で す。

同様に、異なるサンプル時間のブロックを含んだサブシステムブロックもまた、 サブシステムプロックに関係するシングルレートがないため、ハイブリッドとし て色分けされます。サブシステムのすべてのブロックがシングルレートで流れて いる場合は、サブシステムブロックは、そのレートに対応した色になります。

ブロックの接続

あるブロックの出力端子と他のブロックの入力端子を、それらのブロック間にラ インを描画することにより、接続することができます。ラインは、モデルから発 生してブロック間を移動する信号の経路を表わします。信号については、4-30 ページの"信号の利用"を参照してください。この節の残りの部分では、ブロック 間でのラインの描画方法を説明します。

ブロック間でのラインの描画

つぎのようにして、1つのブロックの出力端子を別のブロックの入力端子に接続します。

 カーソルを最初のブロックの出力端子に合わせます。カーソルを端子に正確 に合わせる必要はありません。カーソルの形は、クロスへアに変わります。



- 2 マウスボタンを押します。
- 3 ポインタを2番目のブロックの入力端子までドラッグします。カーソルは、端子上か端子の近くまたはブロックの中に合わせることができます。ブロックの中にカーソルを合わせると、ラインは最も近い入力端子に接続されます。 カーソルの形は、二重のクロスへアに変わります。



4 マウスボタンを解除します。Simulink は、信号の流れる方向を示す矢印をもつ 接続線で端子記号を置き換えます。ラインは、出力から入力の方向へも、入 力から出力の方向へも引くことができます。矢印は、該当する入力端子に描 かれ、信号も同様です。



Simulink は、水平なラインセグメントと垂直なラインセグメントを用いて接続線を描きます。斜線を引くためには、Shift キーを押しながらラインを引きます。

分岐線の描画

分岐線は、既存のラインを起点とするラインで、ブロックの入力端子にその信号 を伝搬します。既存のラインと分岐線は、いずれも同じ信号を伝達します。分岐 線を用いると、1つの信号を複数のブロックに伝搬することができます

つぎの例で、Product ブロックの出力は、Scope ブロックと To Workspace ブロックに伝搬することができます。



分岐線を追加するには、つぎの手順に従います。

- 1 分岐線の起点となるラインにポインタを合わせます。
- 2 Ctrl キーを押しながら、左マウスボタンを押します。
- ポインタをターゲットブロックの入力端子までドラッグして、マウスボタン と Ctrl キーを解除します。

Ctrl キーを押しながら左マウスボタンを使用する代わりに、右マウスボタンを使用することもできます。

ラインセグメントの描画

Simulink が自動的にセグメントを描画する場所ではなく、任意の場所に正確にセ グメントをもつラインを描くこともできます。あるいは、接続されるプロックを コピーする前にラインを描くこともできます。いずれの場合も、ラインセグメン トを描画することによって行えます。

ラインセグメントを描画するには、ブロック線図の空いている場所で終わるライ ンを描きます。ラインが接続されていない方の先端に矢印が現れます。別のライ ンセグメントを追加するには、そのセグメントの先端にカーソルをあわせ、別の セグメントを描画します。Simulink は、セグメントを水平線および垂直線として 描画します。斜めのラインセグメントを描画するには、Shift キーを押しながらラ インを描きます。

ラインセグメントの移動

ラインセグメントを移動するには、つぎの手順に従います。

1 移動したいセグメントにポインタを合わせます。



2 左マウスボタンを押したままにします。



3 希望する位置までポインタをドラッグします。



4 マウスボタンを解除します。



入力端子に接続されたセブメントを移動するには、端子の上にポインタを合わ せ、新しい位置までセグメントの端をドラッグします。出力端子に接続されてい るセグメントを移動させることはできません。

ラインのセグメントへの分割

ラインセグメントは、ラインの両先端をオリジナルの位置に置いたまま、2つの セグメントに分割することができます。Simulink は、ラインセグメントとそれら を結合する頂点を作成します。ラインをセグメントに分割するには、つぎの手順 に従います。

1 ラインを選択します。



2 頂点を必要とするラインにポインタを合わせます。



3 Shift キーを押しながら、マウスボタンを押したままにします。カーソルの形 が新しい頂点を囲む円に変わります。



4 ポインタを希望する位置までドラッグします。



5 マウスボタンと Shift キーを解除します。



ラインの頂点の移動

ラインの頂点の移動は、つぎの手順に従います。

 頂点にポインタを合わせ、マウスボタンを押したままにします。カーソルが 頂点を囲む円に変わります。



2 ポインタを希望する位置までドラッグします。



3 マウスボタンを解除します。



ラインにプロックを挿入

ライン上にブロックをドロップすることで、ラインにブロックを挿入することが できます。Simulink は、ブロックをドロップした場所にブロックを挿入します。 1入力1出力のブロックのみが挿入可能です。

ラインにブロックを挿入するためには、つぎのようにします。

1 ブロック上にポインタを合わせ、左マウスボタンを押します。



2 ブロックを挿入したいライン上にブロックをドラッグします。



3 ライン上にブロックをドロップするためにマウスボタンを離します。Simulink はドロップした場所にブロックを挿入します。



信号の利用

この節では、Simulink 信号の概要と、信号の接続の指定、表示、有効性のチェックの方法を説明します。

信号について

信号は、モデルのシミュレーション時に Simulink ブロックの出力に現れる値の ストリームです。信号は、モデル線図内のブロックを接続するラインに沿って移 動すると考えると便利です。しかし、Simulink モデルのラインは、プロック間の 論理的な接続を表わし、物理的な接続を表わさないことに注意してください。そ のため、Simulink の信号と電気信号との類似は完全ではありません。たとえば、 電気信号は、ワイヤを通過するのに時間がかかります。反対に Simulink プロッ クの出力は、接続するブロックの入力に同時に現れます。

信号の大きさ

Simulink ブロックは、1次元または2次元の信号を出力することができます。1 次元 (1-D) 信号は、シミュレーション時間ステップごとに1つの配列 (ベクトル)をもつ単一周波数の1次元 (1-D) 配列出力のストリームで構成されます。2次元 (2-D) 信号は、ブロックのサンプル時間ごとに1つの2-D 配列 (行列)をもつ 単一周波数の2次元配列のストリームで構成されます。Simulink ユーザインタ フェースとドキュメントは、一般に1-D 信号をベクトル、2-D 信号を行列 として 参照します。1要素配列は、しばしばスカラ として参照されます。*1 行ベクトル* は、1行2次元 (2-D) 配列です。*1 行ベクトル*は、1行の2次元 (2-D) 配列です。

Simulink ブロックは、シミュレーション中に受け取ったり出力することが可能な 信号の次元を変化させます。ブロックの中には、任意の次元の信号を受け取り、 出力できるものがあります。スカラまたはベクトル信号のみを受け取り、出力で きるものもあります。特定のブロックの信号の次元を決定するためには、第9章 ,"ブロックリファレンス."のブロックの説明を参照してください。非スカラ信号 の出力が可能なブロックに対して出力信号の次元を決定するものについては、 4-34 ページの"出力信号の大きさの決定"を参照してください。

信号のデータタイプ

データタイプは、内部的に信号の値を表わすために用いるフォーマットを参照します。Simulink 信号のデータタイプは、デフォルトで double です。しかし、他のデータタイプの信号を作成することができます。Simulink は、MATLAB と同じ範囲のデータタイプをサポートします。詳細は、4-45ページの"データタイプの機能"を参照してください。

複素数値信号

Simulink 信号の値は、複素数の場合があります。値が複素数である信号は、複素 数値信号と呼ばれます。複複素数値信号の作成と操作に関する情報は、4-37 ペー ジの"複素数値信号の利用"を参照してください。

バーチャル信号

バーチャル信号とは、信号を視覚的に表現するための信号です。Mux ブロック やサブシステムブロックのようなバーチャルブロック(4-10 ページの"バーチャ ルブロック"を参照)がバーチャル信号を作り出します。バーチャルブロックと 同様に、バーチャル信号もモデルの見た目が単純になるようにするためのもので す。たとえば Mux ブロックを使うと、たくさんの非バーチャル信号(つまり、 非バーチャルブロックを起源とする信号)を、1つのバーチャル信号に減らすこ とができ、モデルを理解しやすくなります。バーチャル信号は、複数の信号を1 つに束ねて1本にまとめた信号と考えることができます。

バーチャル信号は、真に視覚的な存在です。数学的、物理学的な重要性は一切あ りません。ですから、モデルをシミュレーションする時は無視されます。

Simulink は、モデルを実行したり更新するたびに、信号伝播(signal propagation) として知られる手順を用いて、バーチャル信号として表現されている非バーチャ ル信号を決定します。モデルを実行するとき、Simulink は、バーチャル信号に対 応する非バーチャル信号をブロック信号伝搬から見つけて使用します。たとえ ば、つぎのモデルでは、



信号 s4 は、Gain ブロック G1 を通っているように見えます。しかし、s4 はバー チャル信号です。実際に Gain ブロック G1 を通る信号は、s1 です。Simulink は、 モデルを更新したり実行するときに常に自動的にこれらを決定します。

Simulink の Show Propagated Signals オプション (4-41 ページの"信号プロパティ ダイアログボックス"を参照)は、バーチャル信号のラベルに、そのバーチャル



信号が表現している非バーチャル信号のラベルも一緒に表示するオプションで

す。

注意 バーチャル信号は、非バーチャル信号と同じようにバーチャル信号も表現 することができます。たとえば、Mux ブロックを使って、複数のバーチャル信 号と非バーチャル信号を1本のバーチャル信号にまとめることが可能です。 Simulink は、信号伝播の間に、バーチャル信号を構成する信号の中に、バーチャ ル信号があることが判定されると、信号伝播を使って、その中の非バーチャルな 信号を割り出します。この仮定は、Simulink がすべてのバーチャル信号の非バー チャルな構成信号を割り出すまで続けられます。

バス信号

バス選択モード(バス選択モードに関する詳細は、9-53 ページの "Demux" を参照 してください)で、Mux,Demux ブロックを使用し、バス信号を作成することが できます。



バス信号は、信号の集合をあらわすバーチャル信号です。1つにまとめられたワ イヤーの束に似ています。Simulink は、バス信号を表示するために、特別なライ ンスタイルを使用しています。また、Simulink の Format メニューから Signal Dimentions を選択すると、Simulink は、バスを構成する信号の数を表示します。

信号に関する用語集

つぎの表は、Simulink ユーザインタフェースおよびドキュメントにおいて、信号 を説明するために用いる用語をまとめたものです。

用語	意味
複素数値信号	値が複素数である信号
データタイプ	内部的に信号の値を表わすために用いるフォーマット。詳細は、4-45 ページの"データタイプの機能"を 参照。
行列	2次元信号配列
実数値信号	値が実数値 (複素数の反対として) である信号
スカラ	1 要素配列。すなわち、1 要素、1-D あるいは 2-D 配 列
バス信号	Mux や Demux ブロックで作られる信号
信号の伝播	データタイプ、ラベル、サンプル時間、次元のよう な信号やブロックの属性を決めるために Simulink が 使用するプロセスで、相互の接続性によって決めら れる。
サイズ	信号に含まれる要素の数。行列 (2-D) 信号のサイズ は、一般に M x N として表わされます。ここで、M は列数で、N は信号を構成する行数です。
ベクトル	1次元信号配列
幅	ベクトル信号のサイズ
バーチャル信号	もう一つの信号や、信号の集合をあらわす信号。

出力信号の大きさの決定

ブロックが非スカラ信号を発生することができる場合は、ブロックが出力する信号の大きさは、ブロックが source ブロックである場合はブロックのパラメータ に依存します。そうでない場合は、出力の大きさは、ブロックの入力の大きさに 依存します。

Source ブロックの出力の大きさの決定

source ブロックは、入力をもたないブロックです。source ブロックの例には、 Constant ブロックと Sine Wave ブロックが含まれます (Simulink の source ブロッ クの一覧は、表 9-1, Sources Library ブロック を参照してください)。source ブ ロックの出力の次元は、ブロックの Interpret Vector Parameters as 1-D パラメータ が off(ブロックのパラメータダイアログボックスがチェックされていない)の場 合は、出力値のパラメータの大きさと同じです。Interpret Vector Parameters as 1-D パラメータが on の場合は、出力の大きさは、パラメータの大きさが N × 1 ま たは 1 × N である場合を除いて、出力値のパラメータの大きさと同じです。後 者の場合、ブロックは幅 N のベクトル信号を出力します。

source ブロックの出力値パラメータと Interpret Vector Parameters as 1-D パラメー タが出力の大きさを決定する方法の例として、Constant ブロックを考えます。こ のブロックは、Constant value パラメータと同じ定数信号を出力します。つぎの 表は、Constant value パラメータの大きさと、Interpret Vector Parameters as 1-D パ ラメータの設定がどのようにブロックの出力の大きさを決定するかを示していま す。

Constant value	Interpret vector parameters as 1-D	出力
2-D スカラ	off	2-D スカラ
2-D スカラ	on	1-D スカラ
1 × N 行列	off	1 × N 行列
1 × N 行列	on	N 要素ベクトル
N × 1 行列	off	N × 1 行列
N × 1 行列	on	N 要素ベクトル

Constant value	Interpret vector parameters as 1-D	出力
M × N 行列	off	M × N 行列
M × N 行列	on	M × N 行列

Simulink source ブロックを使って、それらが出力する信号の次元を指定することができます。そのため、モデル内で異なる次元の信号を導入することができます。

非 Source ブロックの出力の大きさの決定

ブロックが入力をもつ場合、出力の大きさは、入力の大きさと等しくなります(つぎの節で説明するように、すべての入力は、同じ大きさでなければなりません)。

信号とパラメータの大きさの規則

ブロックが入力をもつ場合、出力の大きさは、入力の大きさと等しくなります(つぎの節で説明するように、すべての入力は、同じ大きさでなければなりません)。

入力信号の大きさの規則

一般に、ブロックへのすべての入力は、同じ大きさでなければなりません。

しかし、ブロックは、すべての非スカラ入力が同じ大きさである限り、スカラと 非スカラ入力が混在することができます。Simulink は、一般的な規則を守るため に、スカラ入力を非スカラ入力と同じ大きさをもつように拡張します(4-36 ペー ジの"入力のスカラ拡張"を参照)。

ブロックパラメータの大きさの規則

一般に、ブロックのパラメータは、対応する入力と同じ大きさでなければなりま せん。

一般的規則には2つの例外があります。

ブロックは、非スカラ入力に対応するスカラパラメータをもつことができます。この場合、Simulinkは、一般的な規則を守るために、スカラパラメータを入力と同じ大きさになるように拡張します(4-37 ページの"パラメータのスカラ拡張"を参照)。

 入力がベクトルの場合は、対応するパラメータは、Nx1 または 1xN 行列です。 この場合、Simulink は、入力ベクトルの対応する要素に N 行列要素を適用し ます。この例外を使って、MATLAB の行ベクトルまたは列ベクトルを使うこ とができます。これらはそれぞれ実際は 1xN または Nx1 の行列で、ベクトル 入力に適用するパラメータを指定します。

ベクトルあるいは行列入力変換規則

Simulink は、次の状況下でベクトルを行あるいは列行列に、また、行または列行列をベクトルに変換します。

- ベクトル信号が、行列を必要とする入力に接続される場合、Simulink は、ベクトルを1行、あるいは、1列の行列に変換します。
- 1列あるいは1行の行列が、ベクトルを必要とする入力に接続される場合、 Simulink は行列をベクトルに変換します。
- ブロックへの入力が、ベクトルと行列の混在で、行列入力がすべて1列あるいは1行の場合、Simulinkは、それぞれベクトルを1列あるいは1行の行列に変換します。

注意 ベクトルや行列の変換が、シミュレーション中に起こった時に、ワーニン グやエラーメッセージを表示するように設定することができます。詳細は、5-27 ページの"設定オプション"を参照してください。

入力とパラメータのスカラ拡張

スカラ拡張とは、スカラ値を同一要素のベクトルに変換することです。Simulink は、ほとんどのブロックに対する入力またはパラメータにスカラ拡張を適用しま す。第9章の"ブロックリファレンス"のブロックの記述には、Simulink がブロッ クの入力およびパラメータにスカラ拡張を適用するかどうかを示しています。

入力のスカラ拡張

入力のスカラ拡張は、非スカラ入力または非スカラパラメータの大きさと一致さ せるためのスカラ入力の拡張を意味します。ブロックへの入力においてスカラ信 号と非スカラ信号が混在する場合は、Simulink はスカラ入力を非スカラ入力と同 じ大きさをもつ非スカラ信号に拡張します。拡張された信号の要素は、信号が拡 張されたスカラの値と等しくなります。
つぎのモデルでは、入力のスカラ拡張を説明します。このモデルは、スカラとベ クトルの入力を加えます。Constant1 ブロックからの入力は、Constant ブロック からのベクトル入力のサイズと一致するようにスカラ拡張されます。入力はベク トル [3 3 3] に拡張されます。



ブロックの出力がパラメータの関数で、パラメータが非スカラであるとき、 Simulink はパラメータの大きさと一致させるためスカラ入力を拡張します。たと えば、Simulink は非スカラゲインパラメータの大きさと一致させるため、スカラ 入力を Gain ブロックに拡張します。

パラメータのスカラ拡張

ブロックが非スカラ入力をもち、対応するパラメータがスカラである場合、 Simulink はスカラパラメータを入力と同じ要素数になるように拡張します。拡張 されたパラメータの各要素は、オリジナルのスカラの値と等しくなります。 Simulink は、その後拡張されたパラメータの各要素を対応する入力要素に適用し ます。

つぎの例は、スカラパラメータ(Gain)がブロック入力のサイズと一致する、等しい値の要素からなるベクトル、すなわち3要素ベクトルに拡張されることを示します。



複素数値信号の利用

デフォルトで、Simulinkの信号の値は実数です。しかしながら、モデルは値として複素数をもつ信号を作成し、操作することができます。

つぎのような方法で、モデルに複素数値の信号を導入することができます。

- ルートレベルに配置された入力端子を利用して MATLAB ワークスペースから モデルに複素数値の信号データを読み込みます。
- モデルに Constant ブロックを用意し、その値を複素数に設定します。

 複素数信号の実数部と虚数部に関連する実数信号を作成し、それから Real-Imag to Complex ブロックを使って、複素数信号として結合します。

複素数信号を受け入れるブロックによりそれらの信号を操作します。ほとんどの Simulink ブロックは、入力として複素数信号を受け入れます。ブロックが複素数 信号を受け入れるかどうかわからない場合、第9章,"ブロックリファレンス."で プロックに対するドキュメントを参照してください。

信号の接続のチェック

多くの Simulink ブロックは、受け取ることが可能な信号のタイプに制限があり ます。モデルのシミュレーションの前に、Simulink は、すべてのブロックを チェックして、それらが接続する端子によって信号出力のタイプを適合できるこ とを確認します。不整合がある場合は、Simulink は、エラーを送信し、シミュ レーションを停止します。シミュレーションの実行前にそのようなエラーを検出 するためには、Edit メニューから Update Diagram を選択してください。 Simulink は、プロック線図のアップデート処理中に発見された不正な接続を報告 します。

信号の表示オプションの設定

Simulink は、ブロック線図上で信号の特性を表示するためにつぎのようなオプ ションを提供しています。

信号の表示オプション	説明
Wide nonscalar lines	ベクトルまたは行列信号を伝達するライン を、スカラ信号を伝達するラインよりも太い 線で描画します。
Signal dimensions	信号を伝達するラインの横に信号の大きさを 表示します。
Port data types	信号を発生する出力端子の横に信号のデータ タイプを表示します。

これらのオプションは、Simulink の Format メニューまたはモデルのコンテキスト(右クリック)メニューによって設定することができます。

信号の名前

つぎに示す方法によって、信号に名前を付けることができます。

- 信号のラベルの編集
- 信号のプロパティダイアログのNameフィールドを修正(4-41ページの"信号プロパティダイアログボックス"参照)
- 信号を表わしている端子やラインの name パラメータを設定します。たとえば、 p = get param(gcb, 'PortHandles')

 $l = get_param(p.Inport, 'Line')$

set_param(l, 'Name', 's9')

信号ラベル

信号ラベルは信号の名前を表示します。バーチャル信号のラベルは、オプション で、鍵括弧 (<) で表現される信号を表示します。信号のラベルは編集することが でき、信号の名前が変更されます。

信号ラベルを作成するには、ラインセグメントをダブルクリックし、挿入点にラ ベルを入力します。モデルの別の部分をクリックすると、ラベルはその位置に固 定されます。

注意 信号ラベルを作成する場合は、必ずラインの上をダブルクリックするよう に注意してください。ラインの近くの空いている領域をクリックすると、信号ラ ベルではなくモデル注釈が作成されます。

ラベルは、水平のラインまたはラインセグメントの上か下、および垂直のライン またはラインセグメントの左か右に表示することができます。ラベルの位置は、 どちらかの端、中央、またはそれら3つの位置の任意の組み合わせが可能です。

信号ラベルを移動するには、ラベルをライン上の新しい位置にドラッグします。 マウスボタンを解除すると、ラベルはラインの近くの位置に固定されます。

信号ラベルをコピーするには、Ctrl キーを押したままラベルをライン上の別の位 置にドラッグします。マウスボタンを解除すると、ラベルが元の位置と新しい位 置に現れます。

信号ラベルを編集するには、それを選択します。

- ラベルを置き換えるには、ラベルをクリックした後、ダブルクリックするか カーソルをドラッグしてラベル全体を選択します。それから、新しいラベル を入力します。
- キャラクタを挿入するには、2つのキャラクタの間をクリックして挿入点を決定して、テキストを挿入します。
- キャラクタを置き換えるには、マウスをドラッグして置き換えるテキストの 範囲を選択し、新しいテキストを入力します。

すべての信号ラベルを削除するには、ラベル内のすべてのキャラクタを削除しま す。ラベルの外側をクリックすると、すべてのラベルが削除されます。1つのラ ベルのみを削除するには、Shift キーを押しながらラベルを選択し、Delete キーか Backspace キーを押します。

信号ラベルのフォントを変更するには、信号を選択し、Format メニューから Fontを選んで Set Font ダイアログボックスからフォントを選択してください。

バーチャル信号で表わされる信号の表示

バーチャル信号で表現される信号を表示するには、信号のラベルをクリックした後、信号の名前の後ろに小なり記号(<)を入力します(信号に名前がない場合は、小なり記号だけを入力します)。信号ラベルの外側の任意の場所をクリックします。Simulinkは、ラベル編集モードを終了し、ラベル内の括弧の中のバーチャル信号を表示します。

また、信号のプロパティダイアログの Show Propagated Signals オプションを選択 しても、バーチャル信号によって表現される信号を表示することができます (4-41 ページの"信号プロパティダイアログボックス"を参照してください)。

信号プロパティの設定

信号はプロパティをもっています。信号のプロパティを確認したり設定するため に、Simulink の Signal Properties ダイアログを利用します。ダイアログを表示す るために、信号を伝播するラインを選択し、Simulink の Edit メニューから Signal Properties を選択します。

信号プロパティダイアログボックス

Signal Properties ダイアログを利用して、信号プロパティを確認したり編集したりできます。

🛿 Signal Properties: a 👘 🗌	_ 🗆 🗡
CDocumentation	
Signal name: Show propagated signals: on	<u> </u>
a	
Description:	
Document link:	
	_
· · · · · · · · · · · · · · · · · · ·	
Signal monitoring and code generation options	
Displayable (Test Point)	
RTW storage class: Auto	
BTW storage type qualifier:	
	_
OK Cancel Help A	pply

ダイアログは、つぎのコントロールをもっています。

Signal Name 信号の名前

信ちの名削

Show propagated signals

注意 このオプションは、バーチャルブロックから始まる信号にだけ表示されま す。

伝播される信号の名前を表示します。つぎのオプションから選択できます。

オプション	説明
off	信号ラベル内のバーチャル信号によって表現される信号を表 示しない。
on	信号ラベル内のバーチャル信号によって表現されるバーチャ ル信号と非バーチャル信号を表示する。たとえば、バーチャ ル信号 s1 は、非バーチャル信号 s2 とバーチャル信号 s3 を表 現していると仮定します。このオプションが選ばれると、s1 のラベルは s1 <s2,s3> となります。</s2,s3>
all	バーチャル信号が、直接あるいは間接的に表現するすべての 非バーチャル信号を表示します。たとえば、バーチャル信号 s1 は、非バーチャル信号 s2 とバーチャル信号 s3 を表現し、 バーチャル信号 s3 は、非バーチャル信号 s4 と s5 を表現して いると仮定します。このオプションが選択されると、s1 のラ ベルは s1 <s2,s4,s5> となります。</s2,s4,s5>

Description

このフィールドに信号に関する記述を入力します。

Document link

フィールドに、信号に対するドキュメントを表示する MATLAB 表記を入力しま す。ドキュメントを表示するために、フィールドラベル(つまり、"ドキュメン トリンク")をクリックします。たとえば、つぎの表記

web(['file:///' which('foo_signal.html')])

をフィールドに入力すると、フィールドラベルをクリックした時に MATLAB の デフォルトの Web ブラウザが起動し foo_signal.html を表示します。

Displayable (Test Point)

このオプションをクリックすると、シミュレーション中に信号を表示可能である ことを示します。 **注意** つぎの2つのコントロールは、Real-Time Workshop でモデルからコードを 生成する時に使用されるプロパティを設定します。モデルからコードを生成しな い場合は、無視してください。

RTW storage class

リストからこの信号のストレージクラスを選択します。リストされたオプション の説明は、*Real-Time Workshop User's Guide* を参照してください。

RTW storage type qualifier

リストからこの信号のストレージタイプを選択します。詳細は、*Real-Time Workshop User's Guide* を参照してください。

注釈

注釈は、モデルに関するテキスト情報を提供します。注釈は、ブロック線図の空 いている領域に追加することができます。



モデル注釈を作成するには、ブロック線図の空いている領域をダブルクリックし ます。小さな長方形が現れ、カーソルの位置が挿入点になります。注釈内容を入 力します。各行は、注釈を囲む長方形内部でセンタリングされます。

注釈を移動するには、それを新しい位置までドラッグします。

注釈を編集するには、それを選択します。

- Microsoft Windows または UNIX システム上で注釈を置き換えるには、注釈をク リックした後で、ダブルクリックするかカーソルをドラッグしてそれを選択 します。それから、新しい注釈を入力します。
- キャラクタを挿入するには、2つのキャラクタの間をクリックして挿入点を決め、テキストを挿入します。
- キャラクタを置き換えるには、マウスをドラッグして置き換えるテキストの 範囲を選択し、新しいテキストを入力します。

注釈を削除するには、Shift key を押したまま注釈を選択し、Delete キーまたは Backspace キーを押します。

注釈のすべてまたは一部のフォントを変更するには、変更したい注釈内のテキストを選択し、Format メニューから Font を選択します。ダイアログボックスからフォントとサイズを選択します。

注釈のテキスト配置 (たとえば、左、中央、右)を変更するには、注釈を選択 し、モデルウィンドウの Format か、コンテキストメニューから Text Alignment を選択します。そして、Text Alignment サブシステムから整列オプションの1つ (たとえば Center)を選択します。

データタイプの機能

用語"データタイプ"は、コンピュータがどのような方法でメモリ上の数を表現 するかを言及します。データタイプは、数に割り当てられるストレージの量を決 定し、2進数の型として数値を符号化するために利用される理論体系を決定し、 型を操作するために有効な演算を決定します。ほとんどのコンピュータは、数を 表現するためのデータタイプを選択することができ、精度とダイナミックレン ジ、性能、メモリ使用量の面で明確な利点を持っています。MATLAB プログラ ムの性能を最適化するためのデータタイプの利点を有効にするために、 MATLAB では MATLAB 変数のデータタイプを指定することができます。 Simulink は、Simulink 信号とブロックパラメータのデータタイプを指定すること ができ、上記の能力を持っています。

モデルの信号とブロックパラメータのデータタイプを指定することは、特に実時 間制御アプリケーションに有効です。たとえば、Simulink モデルは、The MathWorks 社が提供している Real-Time Workshop のような自動コード生成ツー ルを利用してモデルから生成されたコードにおいて、信号やブロックパラメータ を表現するために利用する最適なデータタイプを指定することができます。モデ ルの信号とパラメータに対して最も適切なデータタイプを選択することによっ て、性能を劇的に向上させ、モデルから生成されるコードのサイズを小さくする ことができます。

Simulink は、シミュレーションの実行前や実行中に、モデルがタイプセーフで ある、つまり、モデルから生成されたコードがオーバーフローやアンダーフロー せず正確な結果を出力することを補償するための検査を広範囲にわたって実行し ます。Simulinkのデフォルトのデータタイプ (double)を利用している Simulink モ デルは、本質的にタイプセーフです。従って、モデルからコード生成をしたり、 デフォルト以外のデータタイプを利用する予定がない場合、この節の残りは読み 飛ばして構いません。

一方、モデルからコードを生成したり、デフォルト以外のデータタイプを利用す るつもりであれば、この節の残り、特にデータタイプの法則に関する節(4-48 ページの"データタイプの法則"を参照してください)を注意深く読んでください。これにより、データタイプエラーによってモデルの完成までの実行やシミュ レーションができなくなることを避けることができます。

Simulink でサポートされるデータタイプ

Simulink は、全ての組み込み MATLAB データタイプをサポートします。用語 " 組み込みデータタイプ "は、MATLAB ユーザが定義したデータタイプに対する ものとして、MATLAB 自身で定義されるデータタイプを言及します。明確に記 述されていない場合、Simulink ドキュメントにおける用語 " データタイプ " は、 組み込みデータタイプのことです。つぎの表は、MATLAB の組み込みデータタ イプ一覧です。

名前	詳細
double	倍精度浮動小数点
single	単精度浮動小数点
int8	符号付き8ビット整数
uint8	符号なし8ビット整数
int16	符号付き 16 ビット整数
uint16	符号なし 16 ビット整数
int32	符号付き 32 ビット整数
uint32	符号なし 32 ビット整数

Simulink は、組み込みタイプに加えて、boolean (1 または 0) タイプを定義でき、 内部的には uint8 の値で表現されます。

ブロックがサポートするデータと数の信号タイプ

全ての Simulink ブロックは、デフォルトで double 型の信号を受け入れます。ブ ロックの中には、boolean 型の入力を必要とするものや、多数のデータタイプの 入力をサポートするものがあります。パラメータと入力値および出力値に対して 特定のブロックがサポートするデータタイプに関する情報は、第9章の"ブロッ クリファレンス"を参照してください。ブロックに対するドキュメントが、デー タタイプを特定していない場合は、ブロックの入力または出力は double タイプ のみです。

ブロックパラメータのデータタイプの指定

データタイプがユーザが指定可能であるブロックパラメータを入力する場合は、 つぎの書式

type(value)

を使って、パラメータを指定します。ここで、type はデータタイプ名で、value はパラメータ値です。つぎの例は、この書式を説明しています。

single(1.0)	単精度値 1.0 を指定
int8(2)	8 ビット整数 2 を指定
int32(3+2i)	実部と虚部が 32 ビット整数である複素数値を指定

特定のデータタイプの信号の作成

特定のデータタイプをもつ信号をつぎの方法でモデル内に導入することができます。

- ルートレベルの inport または From Workspace ブロックにより、希望するタイプ の信号データを MATLAB ワークスペースからモデルにロードします。
- モデル内に Constant ブロックを作成し、パラメータを希望するタイプに設定します。
- Data Type Conversion ブロックを使って信号を希望するデータタイプに変換します。

端子のデータタイプの表示

モデル内の端子のデータタイプを表示するには、SimulinkのFormat メニューから Port Data Types を選択します。Simulink は、図の要素のデータタイプを変更しても、端子のデータタイプ表示を更新しません。表示を更新するには、Ctrl-Dを実行します。

データタイプの伝達

シミュレーションを実行したり、端子のデータタイプの表示を有効にしたり、端 子のデータタイプの表示を更新したときには、Simulink はデータタイプの伝達の 手順を実行します。この手順では、型が特に指定されていない信号の型を決定 し、信号と入力端子の型が矛盾しないことを補償するための検査をします。型の 矛盾があると、Simulink は、データタイプが矛盾している信号と端子を特定する エラーダイアログを表示します。Simulink はまた、型が矛盾する信号のパスを八 イライトします。 **注意** 矛盾を解決するために、モデルに typecasting (データタイプの変換)ブロックを挿入することができます。詳細は、4-49ページの"信号の型変換"を参照してください。

データタイプの法則

つぎの法則を確認することは、タイプセーフなモデルを作成する手助けになり、 その結果、エラー無しで実行することができるでしょう。

- 一般的に信号のデータタイプはパラメータのデータタイプに影響せず、逆のことも言えます。
 この法則における重要な例外は、出力のデータタイプがパラメータのデータタイプによって決定される Constant ブロックです。
- ブロックの出力が入力とパラメータの関数で、入力とパラメータが異なる データタイプの場合、Simulinkは出力を計算する前にパラメータを入力の型に 変換します。

詳細に関しては、4-49ページの"パラメータの型変換"を参照してください。

- 一般的に、ブロックは入力のデータタイプで出力します。
 重要な例外は、Constant ブロックと、出力のデータタイプがブロックパラメータによって決定される Data Type Conversion ブロックです。
- バーチャルブロックは、その入力に関してあらゆる型の信号を受け入れます。
 バーチャルブロックの例は、Mux ブロックと Demux ブロック、条件無しで実行されるサブシステムです。
- ノンバーチャルブロックの端子に接続される信号ベクトルの要素は、同じ データタイプでなければいけません。
- ノンバーチャルブロックの入力データ端子に接続される信号は、同じ型でなければいけません。
- コントロール端子(たとえば、イネーブル端子とトリガ端子)は、boolean型や double型の信号を受け入れます。
- Solver ブロックは、double 型の信号のみを受け入れます。
- ブロックに double 型でない信号を接続すると、そのブロックに対するゼロク ロッシングの検出を無効にします。

厳密な Boolean 型の検査を有効にする

Simulink は、boolean 型の入力が望まれるブロックに double 型の信号が接続され ていることを検出すると、デフォルトでは信号のエラーを検出しません。このこ とは、double 型のデータタイプのみをサポートしている以前のバージョンの Simulink で作成されたモデルとの互換性を補償します。Simulation Parameters ダ イアログボックスの Advanced パネル上で Boolean logic signals オプションの チェックを外すと、厳密な boolean 型の検査を有効にすることができます (5-29 ページの"アドバンスドパネル"を参照してください)。

信号の型変換

Simulink は、信号のデータタイプが受け入れられないブロックに接続されている 信号を検出するとエラーを表示します。そのような接続を作成したい場合、信号 の型をブロックが受け入れる型に変換しなければいけません。SimulinkのData Type Conversion ブロックを利用して、そのような変換が実行できます(9-49 ペー ジの "Data Type Conversion" を参照してください)。

パラメータの型変換

一般的に、シミュレーション中に入力信号とパラメータの関数でブロック出力を 計算する時に、Simulink は暗黙のうちにパラメータのデータタイプを信号のデー タタイプに(データタイプが異なっていれば)変換します。この法則にはつぎの 例外があります。

信号のデータタイプがパラメータ値を表現できないとき、Simulink はシミュレーションを中止し、エラーを表示します。

たとえば、つぎのモデルを考えます。



このモデルは、定値入力信号を増幅するためにゲインブロックを利用してい ます。ゲインブロックの出力を計算するために、入力信号とゲインの乗算を 計算することが要求されます。そのような計算は、2つの値が同じデータタイ プであることを必要とします。しかしながら、この例では、信号のデータタ イプ uint8(符号なし8ビットワード)がゲインパラメータのデータタイプ int32(符号付き 32 ビット整数)と異なっています。従って、ゲインブロックの 出力の計算は、型変換を必要とします。

そのような変換を実行するとき、Simulink は通常パラメータの型を信号の型に 変換します。従って、この例では、Simulink がこの変換を実行できるための十 分条件は、入力信号のデータタイプ (uint8) がゲインを表現できることです。 この例では、ゲインが uint8 のデータタイプ (0 ~ 255) の範囲で表現できる 255 なので、Simulink は変換を実行することができます。従って、このモデルはエ ラーなしで実行されます。しかしながら、ゲインがほんのわずかでも大きい と(たとえば、256)、シミュレーションを実行しようとすると、Simulink は out-of-range のエラーを表示します。

 信号のデータタイプが、精度の劣化なしでパラメータ値を表現できるとき、 Simulink はワーニングメッセージを表示し、シミュレーションを継続します。 (5-27 ページの"設定オプション"を参照してください)。

たとえば、つぎのモデルを考えてください。



この例では、ゲインの値が分数成分をもつのに対して、信号の型は整数値に のみ適応します。このモデルをシミュレーションすると、Simulink はゲインを 最も近い整数値(2)に切り捨て、精度の劣化の警告を出力します。一方、ゲイ ンが 2.0 のとき、変換に精度の劣化が必要とされないので、Simulink は警告無 しでモデルをシミュレーションすることができます。

注意 int32 のパラメータの float や double への変換は、精度の劣化を伴います。 劣化は、パラメータ値の大きさが大きい場合に顕著になります。int32 のパラ メータ変換が精度の劣化を伴わないとき、Simulink はワーニングメッセージを表 示します。

データオブジェクトの機能

Simulink データオブジェクトを使用することで、Simulink モデルが使うデータ情報(つまり、信号やパラメータ)を指定したり、モデルにデータ情報そのものを保存することができます。Simulink は、パラメータが変更可能かを決定づけたり、信号の可視性を決めたり、コードを生成するためにデータオブジェクトのプロパティを使用します。また、データオブジェクトを使用すると、パラメータの最小値や最大値のような、シミュレーションの補正に重要な情報を指定することができます。さらに、データオブジェクトはモデルと共に保存することができます。このように Simulink では、内蔵タイプのモデルを作ることができます。

データオブジェクトクラス

データオブジェクトは、データオブジェクトクラスから呼ばれる1つのオブジェクトのインスタンスす。データオブジェクトクラスは、インスタンスのプロパティや、インスタンスの作り方や扱い方のメソッドを定義します。Simulink は、2つの組み込みデータクラス、Simulink.ParameterとSimulink.Signalをもち、各々パラメータと信号データオブジェクトを定義します。

データオブジェクトプロパティ

データオブジェクトプロパティは、データの項目の値やストレージタイプのよう に、オブジェクトが記述するデータ項目の属性を指定します。すべてのプロパ ティは名前と値をもちます。値はプロパティによって配列や構造体にすることが できます。

データオブジェクトパッケージ

Simulink は、パッケージと呼ばれるクラスの集合体の中にクラスを保持します。 Simulink は、デフォルトで Simulink という名前の単一のパッケージを持ちます。 Simulink のクラス、Simulink.Parameter と Simulink.Signal は、Simulink パッケージに 属します。ユーザは、その下に追加のパッケージを作成し、それらのクラスに属 すクラスを定義することができます。

クオリファイドネーム

MATLAB のコマンドラインや M- ファイルプログラムからクラスを参照するに は、つぎのように"ドット"を使って、クラスの名前とクラスパッケージの名前 の両方を指定してください。

PackageName.ClassName

PackageName.ClassNameの表記は、クラスのクオリファイドネームと呼ばれています。たとえば、Simulink パラメータクラスのクオリファイドネームは、Simulink.Parameter です。

2 つのパッケージでは、クラス名は同じだけれども、別個のクラスを持つことが できます。たとえば、パッケージAとBは、両方ともCという同じクラス名を 持つことができます。それぞれのクオリファイドネームを使用して、MATLAB コマンドラインや M-ファイルプログラムからクラスを参照することができま す。クラスを作成するときに、パッケージによって名前の一致を避けることがで きます。たとえば、類似した Simulink のクラス名が一致しないようにと悩まな くても Parameter と Signal クラスを作成することができます。

注意 クラスとパッケージの名前は大文字、小文字を区別します。たとえば、同 じクラスを参照するときに A.B と a.b を同じ物として使用することはできませ ん。

データオブジェクトの作成

Simulink データオブジェクトを作成するには、Simulink データエクスプローラ か、または MATLAB コマンドのどちらかを使用します。データエクスプローラ を使用してデータオブジェクトを作成する場合の情報は、4-61 ページの "Simulink データエクスプローラ"を参照してください。

MATLAB コマンドラインやプログラム内からデータオブジェクトを作成するには、つぎのコマンドを使用してください。

h = package.class(arg1, arg2, ...argn);

ここで、h は MATLAB 変数、package はクラスが属するパッケージの名前、class はクラスの名前、そして arg1, arg2, ... argn はオブジェクトのコンストラクタに渡さ れるオプショナルの引数です (Simulink.Parameter や Simulink.Signal クラスのコンス トラクタは、引数をもちません)。たとえば、Simulink.Parameter クラスのインスタ ンスを作成するには、MATLAB コマンドラインに、

hGain = Simulink.Parameter;

と入力してください。

このコマンドは、Simulink.Parameter のインスタンスを作成し、ゲインにそのハンドルを保存します。

データオブジェクトプロパティへのアクセス

データオブジェクトプロパティを設定したり取得するには、Simulink データエク スプローラ (4-61 ページの "Simulink データエクスプローラ"を参照)や、 MATLAB コマンドを使用します。オブジェクトプロパティを取得したり設定す るデータエクスプローラの使用方法に関する詳細は、4-58 ページの"パッケージ の作成"を参照してください。

MATLAB コマンドラインや M-ファイルプログラムからデータオブジェクトの プロパティにアクセスするには、つぎの構文を使用します。

hObject.property

ここで、hObject は、オブジェクトのハンドルで、property はプロパティの名前 です。たとえば、つぎのコード

hGain = Simulink.Parameter; hGain.Value = 5;

は、Simulink ブロックパラメータオブジェクトを作成し、オブジェクトのプロパ ティの値に 5 を設定します。たとえば、gain.RTWInfo.StorageClass は gain パラメー タの StorageClass プロパティを戻します。

データオブジェクトメソッドの呼び出し

つぎの構文を使用して、データオブジェクトメソッドを呼び出します。

hObject.method

あるいは

method(hObject)

ここで、hObject はオブジェクトのハンドルです。Simulink は、データオブジェクトのために、つぎのメソッドを定義します。

• get

MATLAB 構造体としてオブジェクトプロパティを戻します。

• copy

オブジェクトのコピーを作成し、コピーにハンドルを戻します。

データオブジェクトの保存とロード

データオブジェクトを MAT-ファイルに保存するには、MATLAB の save コマン ドを使います。そのデータオブジェクトを後のセッションで MATLAB ワークス ペースに復帰させるには、MATLAB の load コマンドを使用します。保存したオ ブジェクトのクラスの定義を復帰するには、MATLAB パスが通っているところ に置かなければなりません。保存したオブジェクトのクラスが、オブジェクトを 保存した後に、新しいプロパティを取得した場合、Simulink は復帰させたオブ ジェクトに新しいプロパティを付加します。オブジェクトを保存した後に、クラ スがプロパティを失った場合、Simulink は残っているプロパティのみ復帰しま す。

Simulink モデル内のデータオブジェクトの使用

パラメータや信号として Simulink モデルのデータオブジェクトを使用すること ができます。パラメータや信号としてデータオブジェクトを使用することで、シ ミュレーションを指定したり、オブジェクトベースのコード生成オプションを指 定することができます。

パラメータとしてデータオブジェクトを使用

ブロックパラメータとして、Simulink.Parameter クラスや子孫クラスのインスタ ンスを使用することができます。ブロックパラメータとして、パラメータオブ ジェクトを使用するには、

- 1 MATLAB コマンドラインか Simulink データエクスプローラで、パラメータオ ブジェクトを作成します。
- オブジェクトの Value プロパティをブロックパラメータの定義したい値に設定します。
- 3 パラメータオブジェクトストレージクラスを設定し、パラメータの変更可能 にするかの決定づけ (4-56 ページの "データオブジェクトクラスの作成"参照) やコード生成オプション(詳細は Real-Time Workshop のマニュアルを参照) を選択するために、プロパティを入力します。
- 4 ブロックパラメータダイアログボックスか、または set_param コマンドを使って、ブロックパラメータとしてパラメータオブジェクトを定義します。

他のセッションでも継続して使えるパラメータオブジェクトを作成する方法については、4-56ページの"継続して使用するパラメータと信号オブジェクトの作成" を参照してください。

パラメータのチューニングを定義するためにパラメータオブジェクトを 使用

インラインパラメータシミュレーションオプションが設定 (5-29 ページの"モデ ルパラメータの設定"参照)されている時でも、パラメータをチューニングした い場合は、パラメータオブジェクトの RTWInfo.StorageClass プロパティを'Auto' (デフォルト)以外の値に設定してください。

gain.RTWInfo.StorageClass = 'SimulinkGlobal';

RTWInfo.StorageClass プロパティを Auto 以外の値に設定した場合は、モデルの チューニングパラメータテーブル (5-32 ページの"モデルパラメータ設定ダイア ログボックス"を参照)の中に、そのパラメータを含めないでください。

注意 モデルパラメータ設定ダイアログボックス内に定義したパラメータのプロ パティとパラメータオブジェクトとして定義したパラメータのプロパティで同じ 名前が検知された場合、Simulink はシミュレーションを停止し、エラーメッセー ジを表示します。

信号としてデータオブジェクトを使用

信号プロパティを指定するために Simulink.Signal クラスや子孫クラスのインスタ ンスを使用することができます。信号プロパティを指定するための信号オブジェ クトとして、データオブジェクトを使用するには、

- 1 モデルのワークスペースに信号データオブジェクトを作成します。
- ストレージクラスを設定し、信号の可視化(4-56ページの"テストポイントを指定するために信号オブジェクトを使用"を参照)やコード生成オプション(コード生成オプション指定のための信号プロパティの使用に関する詳細は、 Real-Time Workshopのマニュアルを参照)を指定するために、信号オブジェクトのプロパティを入力します。
- 3 信号データオブジェクトと同じプロパティを持たせたい信号のラベルを変更し、信号と同じ名前にします。

別の Simulink セッションでも継続して使用できる信号オブジェクトの作成に関 する情報は、4-56 ページの"継続して使用するパラメータと信号オブジェクトの 作成"を参照してください。

テストポイントを指定するために信号オブジェクトを使用

信号をテストポイントとしたい(つまり、シミュレーションの間中、いつでもフ ローティングスコープで表示することができる)場合、対応する信号オブジェク トの RTWInfo.StorageClass プロパティを auto 以外の値に設定してください。

注意 信号オブジェクトによって指定される信号のプロパティと、Signal Properties ダイアロブボックス (4-41 ページの"信号プロパティダイアログボック ス"を参照)で指定されるパラメータのプロパティで同じ名前が検出されると、 Simulink は、シミュレーションを停止し、エラーメッセージを表示します。

継続して使用するパラメータと信号オブジェクトの作成

別の Simulink セッションでも継続して使用するパラメータや信号オブジェクト を作成するには、最初にオブジェクトを作成するスクリプトを記述するか、コマ ンドラインでオブジェクトを作成し、それらを MAT-ファイル (4-54 ページの" データオブジェクトの保存とロード"を参照してください)に保存します。次に、 そのオブジェクトを使用するモデルの PreLoadFcn コールバックルーチンとして load コマンドを使用してください。たとえば、ファイル名が data_objects.mat と いうファイルにデータオブジェクトを保存し、それを適用するモデルがオープン されアクティブな状態にあるとします。そこで、次のコマンド、

set_param(gcs, 'PreLoadFcn', 'load data_objects');

を、MATLAB コマンドラインに入力すると load data_objects がモデルのプリロー ド関数として設定されます。これにより、次にモデルをオープンする時はいつで も、データオブジェクトがモデルワークスペース内にロードされます。

データオブジェクトクラスの作成

新しくデータオブジェクトクラスを作成するには、クラスのインスタンスを構築 したり、インスタント化するための M- ファイルプログラムを記述する必要があ ります。クラスを含んだ新しいパッケージを作成したい場合、ユーザは、新しい パッケージのための M- ファイルコンストラクタも記述しなければなりません。 注意 Simulink デモディレクトリ (matlabroot/toolbox/simulink/simdemos) には、 UserDefined と呼ばれるユーザ定義のデータオブジェクトクラス定義のサンプル が含まれています。このクラス定義は、ユーザ自身のクラスを作成する時のテン プレートとして使うことができます。ユーザ自身のクラスを作成するために、こ のサンプルテンプレートをコピーして修正することができます。

パッケージのディレクトリ構造

ユーザは、規定の構造をもつディレクトリ内に、クラスを定義するプログラムを 保管しなければなりません。

😂 C:\0	nThel	MATLA	BPat	th\@UserD	efined	_ 🗆 ×
<u> </u>	<u>E</u> dit	⊻iew	<u>G</u> o	F <u>a</u> vorites	<u>H</u> elp	
Addres	ss 🚞	C:\OnThe	MATLA	ABPa:h\@Use	Defined	•
C: - 30 - 10 - 10	nTheMA 2015e 50 0 0 0 0 0 0 0 0 0 0 0 0 0	(TLABPat rDefined hema.m Paramete Schema Signal Schema Sgnal n	h m ter.m m			

ディレクトリ構造は、つぎの条件を満たす必要があります。

- 各パッケージは、MATLAB コマンドパス上にパッケージディレクトリと呼ばれるそれ自身のディレクトリをもっていなければなりません。パッケージディレクトリは、@PackageNameという名前にしなければなりません。ここで、PackageNameはパッケージの名前です。
- パッケージ内の各クラスのコードは、クラスディレクトリと呼ばれるパッケージディレクトリの中の別のサブディレクトリ内に存在しなければなりま

せん。クラスディレクトリは @ClassName という名前にしなければなりません。 ここで、ClassName は新しいクラスの名前です。

パッケージディレクトリは、パッケージを構築する schema.m という名前の M-ファイルプログラムを含んでいなければなりません。各クラスディレクトリは、 schema.m という名前のコンストラクタを含み、ClassName.m という名前のインスタ ンス関数を含まなければなりません。ここで、ClassName は新しいクラスの名前 です。

パッケージの作成

パッケージを作成するために、最初に、@package_name という名前のディレクト リを MATLAB パス上のディレクトリに作成します。ここで、@PackageName は、 新しいパッケージの名前です。次に、パッケージディレクトリに schema.m とい う名前の M- ファイルを作成します。schema.m ファイルは MATLAB 関数です。

function schema () % パッケージコンストラクタ関数

schema.package('PackageName');

ここで、PackageName は、新しいパッケージの名前です。

クラスの作成

データオブジェクトクラスを作成するために、

- 新しいクラスを置いておきたいパッケージのディレクトリ内に @ClassName と いう名前のディレクトリを作成します。ここで、ClassName は新しいクラスの 名前です。
- 2 クラスディレクトリの中にクラスコンストラクタを作成します。
- 3 クラスディレクトリの中にクラスインスタンス関数を作成します。

クラスコンストラクタの作成

MATLAB は、クラスディレクトリ内の schema という名前の関数を探して、クラ スのためのコンストラクタを見つけます。そのため、ユーザが作成しているクラ スのクラスディレクトリ内に、この関数を作成しなければなりません。コンスト ラクタは、つぎの例が示すように、create_user_class 関数 (4-60 ページの "create_user_class" を参照してください)を呼び出すことによって、クラスを作成 します。 function schema() % クラスコンストラクタ関数

% 作成するクラスの名前の指定 userClass = 'UserDefined.Parameter';

% ユーザクラスが導出されるクラス名の指定 deriveFromClass = 'Simulink.Parameter';

% このクラスで使用されるユーザ定義一覧表 % のために一般化コンストラクタ関数を呼び出す create_user_enumtype('colors', { 'red', 'green', 'blue'});

% ユーザクラスに含む新しいプロパティを指定する addProperties = { 'UserMATLABArray1', 'MATLAB array', []; ... 'UserMATLABArray2', 'MATLAB array', "; ... 'UserDouble', 'double', 0; ... 'UserInt32', "int32", 0; ... 'off'; ... 'UserOnOff', 'on/off', ": ... 'UserString', 'string', 'UserColorEnum', 'colors', 'red'; ... };

% 一般化クラス作成関数(組込み関数)の呼び出し create_user_class(userClass, deriveFromClass, addProperties);

クラスインスタンス関数の作成

Simulink は、クラスインスタンスを作成するために、クラスインスタンス関数を 使用します。この関数は、クラスと同じ名前の M- ファイルをクラスディレクト リの中から探すことによって、クラスインスタンス関数を見つけます。たとえ ば、クラス名が Parameter の場合、Simulink は Parameter.m という名前の M- ファイ ルと、関数にハンドルを戻す Parameter という名前の関数を含んだ M- ファイルを 探します。最小のインスタンス関数は引数をとらず、単純に、つぎの例が示すよ うな、クラスのためのデフォルトインスタンス関数を呼び出します。

function h = Parameter() % クラスインスタンス関数 % インスタンスクラス h = UserDefined.Parameter; インスタンス関数は、オプションで様々な数の引数をとることができます。つぎ の例題のように、インスタンス関数は、オブジェクトのプロパティを初期化する ために、オプション関数を使用することができます。

function h = Parameter(varargin)

% クラスインスタンス関数

% インスタンスクラス

h = UserDefined.Parameter;

```
% プロパティを初期化(オプション)
if nargin == 1
% 引数が1つだけ与えられる場合、"Value" として扱われる。
h.Value = varargin{1};
end
```

データオブジェクトプロパティの作成

データオブジェクトクラスは、親クラスのプロパティを引き継ぎます。そのコン ストラクタの中に、クラスの追加プロパティを定義することができます。そうす るためには、クラスコンストラクタ関数 (4-60 ページの "create_user_class" を参照) に n × 3 セル配列を渡します。ここで n は、指定されたプロパティの数です。 配列の各行は、名前 (たとえば、'angle')、タイプ (たとえば 'double')、そして対応 するプロパティのデフォルト値を指定します。

Simulink.Signal と Simulink.Parameter クラスは、将来のリリースでは、新しいプロパ ティをとるようになる予定です。したがって、これらのクラスからクラスを引出 す時は、将来、クラスのプロパティの名前と同じ名前にならないようなプロパ ティ名を使用してください。同じ名前を避けるための1つのアプローチ方法は、 あなたの会社の名前を導出されるクラスのプロパティ名に追加することです。

データオブジェクト関数

Simulink は、Simulink データオブジェクトや、Simulink データクラスを作成した り扱うために、つぎの関数を用意しています。

create_user_class. 新しいデータオブジェクトクラスを作成するために、クラスコ ンストラクタファイル (schema.m)の中で、この関数を使用します。この関数は、 3つの引数をとります。

- 新しいクラスのクオリファイドネーム(たとえば、'UserDefined.Parameter')
- 新しいクラスの親のクオリファイドネーム(たとえば、'Simulink.Parameter')

新しいクラスのプロパティを指定するセル配列(4-60ページの"データオブジェクトプロパティの作成"を参照)

create_user_enumtype. この関数は、列挙されたデータタイプを作成するために、 クラスコンストラクタの中で使用します。列挙されたデータタイプとは、有効な 値の集合を伴ったデータタイプのことです。そして、クラスプロパティの1つ以 上のタイプとして、列挙されたタイプを使用することができます。 create_user_enumtype 関数は2つの引数をとります。

- 列挙されたタイプの名前
- このタイプのインスタンスに有効な値の集合を指定したセル配列

たとえば、つぎのコードは、colorsという名前の列挙されたタイプを作成します。

create_user_enumtype('colors', {'red', 'green', 'blue'});

findpackage. パッケージオブジェクトにハンドルを戻します。たとえば、

h_SimulinkPackage = findpackage('Simulink');

findclass. クラスにハンドルを戻します。たとえば、

h_SimulinkParameter = findclass(h_SimulinkPackage, 'Parameter');

findproperty. オブジェクトプロパティにハンドルを戻します。たとえば、

h_ParamValue = findparameter(h_SimulinkParameter, 'Value');

Simulink データエクスプローラ

Simulink データエクスプローラを使って、MATLAB ワークスペース内の変数と データオブジェクトの値を表示したり設定することができます。データエクスプ ローラを開くには、Simulink の Tools メニューから Data explorer を選択するか、 MATLAB プロンプトで、slexplr と入力してください。すると、ダイアログボッ クスが表示されます。

📣 Simulink D	ata Explorer	
Objects		Properties
Nam	e Class	G Simulink.Parameter
🔷 gain	Simulink.Parameter	
Filter option:	double Valid MATLAB variables	Value (1x1 double array)
		Help Close

データエクスプローラは、2つのパネルからなります。左のパネルは、MATLAB ワークスペースに定義されている変数のリストです。表示する変数のタイプを指 定するには、Filter オプションを使用します(たとえば、すべての変数あるいは Simulink データオブジェクトのみなど)。右のパネルは、左のパネルで選択され た変数の値が表示されます。オブジェクトを作成したり、名前を付け直したり削 除するには、左パネルの中でマウスの右ボタンをクリックしてください。プロパ ティの構造体のフィールドを表示するには、プロパティの名前の横にある+ボ タンをクリックしてください。

値を変更するには、valueをクリックしてください。値が文字列の場合は、文字 列を修正してください。プロパティを既に定義されている集合の1つに設定しな ければならないときは、データエクスプローラは、有効な値を表示するドロップ ダウンリストを表示します。そこから値を選択してください。値が配列の場合、 データエクスプローラは、配列の次元やそれぞれの要素の値を設定することがで きる配列エディタを表示します。

Value		×
Enter Expression: Size: 2	by 2	
1 2		
	ок	Cancel

マウスとキーボードの動作のまとめ

以下の表に、ブロック、ライン、信号ラベルを操作するマウスとキーボードの使 い方をまとめます。LMB は左マウスボタン、CMB は中央マウスボタン、RMB は右マウスボタンを押すことを意味します。

最初の表は、ブロックに適用されるマウスとキーボードの動作を示します。

表 4-3: ブロックの操作

作業	Microsoft Windows	UNIX
1 つのブロックを選 択	LMB	LMB
複数のブロックを選 択	Shift + LMB	Shift + LMB; または CMB のみ
つぎのブロックを選 択	Tab	Tab
前のブロックを選択	Shift + Tab	Shift + Tab
別のウィンドウから のブロックのコピー	ブロックのドラッグ	ブロックのドラッグ
ブロックの移動	ブロックのドラッグ	ブロックのドラッグ
ブロックの複製	Ctrl + LMB とドラッグ、 または RMB とドラッグ	Ctrl + LMB とドラッグ、 または RMB とドラッグ
ブロックの接続	LMB	LMB
ブロックの切り離し	Shift + ブロックのドラッ グ	Shift + ブロックのドラッ グあるいは CMB とド ラッグ
選択サブシステムの オープン	Enter	Return
選択サブシステムの 親への移動	Esc	Esc

つぎの表は、ラインに適用されるマウスとキーボードの動作を示します。

表 4-4: ラインの操作

作業	Microsoft Windows	UNIX
ラインを選択	LMB	LMB
複数のラインを選択	Shift + LMB	Shift + LMB; または CMB のみ
分岐線の描画	Ctrl+ラインをドラッグ または RMB とラインを ドラッグ	Ctrl + ラインをドラッグ または RMB + ラインを ドラッグ
ブロック周囲のライ ンの描画	Shift + ラインセグメント の描画	Shift + ラインセグメント の描画、または CMB + セグメントの描画
ラインセグメントの 移動	ラインセグメントの描画	セグメントのドラッグ
頂点の移動	頂点のドラッグ	頂点のドラッグ
ラインセグメントの 作成	Shift + ラインのドラッグ	Shift + ラインのドラッグ または CMB + ラインの ドラッグ

つぎの表は、信号ラベルに適用されるマウスとキーボードの動作を示します。

表 4-5: 信号ラベルの操作

作業	Microsoft Windows	UNIX
信号ラベルの作成	ラインをダブルクリック した後でラベルを入力	ラインをダブルクリック した後でラベルを入力
信号ラベルのコピー	Ctrl+ラベルのドラッグ	Ctrl + ラベルのドラッグ
信号ラベルの移動	ラベルのドラッグ	ラベルのドラッグ

作業	Microsoft Windows	UNIX
信号ラベルの編集	ラベルをクリックした後 で編集	ラベルをクリックした後 で編集
信号ラベルの削除	Shift + ラベルのクリッ ク、それから Delete を押 す	Shift + ラベルのクリッ ク、それから Delete を押 す

表 4-5: 信号ラベルの操作 (Continued)

つぎの表は、注釈に適用されるマウスとキーボードの動作を示します。

作業	Microsoft Windows	UNIX
注釈の作成	ブロック線図内をダブル クリックした後でテキス トを入力	ブロック線図内をダブル クリックした後でテキス トを入力
注釈のコピー	Ctrl+ラベルのドラッグ	Ctrl+ラベルのドラッグ
注釈の移動	ラベルのドラッグ	ラベルのドラッグ
注釈の編集	テキストをクリックし、 編集	テキストをクリックした 後で編集
注釈の削除	Shift + 注釈を選択 , それ から Delete を押す	Shift + + 注釈の選択、そ れから Delete を押す

表 4-6: 注釈の操作

サブシステムの作成

モデルのサイズと複雑さが増すに連れて、ブロックをサブシステムにグループ化 することによってそれを単純化することができます。サブシステムを使用するこ とには、つぎのような利点があります。

- モデルウィンドウに表示されるブロック数を減らすのに役立ちます。
- 機能的に関連するブロックを1つにまとめておくことができます。
- Subsystem ブロックを1つの層に置き、サブシステムを構成するブロックを別の 層に置く、階層的なブロック線図を確立することができます。

サブシステムは、つぎの2つの方法によって作成することができます。

- Subsystem ブロックをモデルに追加した後、そのブロックを開いて、サブシス テムウィンドウに必要なブロックを追加します。
- サブシステムを構成するブロックを追加した後で、それらのブロックをサブ システムにグループ化します。

Subsystem ブロックの追加によるサブシステムの作成

必要なブロックを追加する前にサブシステムを作成するには、Subsystem ブロックをモデルに追加してからサブシステムを構成するブロックを追加します。

- 1 Signals & Systems ライブラリから Subsystem ブロックをモデルにコピーします。
- 2 Subsystem ブロックをダブルクリックして開きます。

Simulink は、ユーザが選んだモデルウィンドウの再利用モード (4-68 ページの "ウィンドウの再利用"参照)により、サブシステムをカレントのウィンドウ か、もしくは新しいモデルウィンドウ上にオープンします。

3 空の Subsystem ウィンドウで、サブシステムを作成します。サブシステムの外部からの入力を示すには Inport ブロックを、外部出力を示すには Outport ブロックを使います。たとえば、つぎのサブシステムには、Sum ブロックと Inport ブロック、Outport ブロックが含まれており、サブシステムへの入力およびサブシステムからの出力を示します。



既存のブロックのグループ化によるサブシステムの作成

サブシステムに変換したいブロックがモデルにすでに含まれている場合、それら のブロックをグループ化することによってサブシステムを作成することができま す。

 サブシステムに含みたいブロックと接続線を境界ボックスで囲みます。ブロックを個々に選択したり Select All コマンドを使用することによって、プロックをグループ化するように指定することはできません。詳細については、 4-8ページの"境界ボックスを用いて複数のオブジェクトを選択する"を参照してください。

たとえば、つぎの図はカウンタを表すモデルを示しています。Sum ブロック と Unit Delay ブロックが境界ボックスで選択されています。



マウスボタンを解除すると、2つのブロックとすべての接続線が選択されます。

2 Edit メニューから Create Subsystem を選択します。Simulink は選択したブロックを Subsystem ブロックで置き換えます。つぎの図は、Create Subsystem コマンドを選択した後(および端子ラベルが読めるように Subsystem ブロックの大きさを変更した後)のモデルを示しています。



Subsystem ブロックをオープンすると、Simulink は下に示すようにその基礎と なっているシステムを表示します。Simulink が、サプシステムの外部からの入力 とサブシステムの外部への出力を表す Inport ブロックと Outport ブロック を追加 する点に注意してください。



すべてのブロックの場合と同様、Subsystem ブロックの名前を変更することがで きます。また、第7章の"マスクを使ったブロックのカスタマイズ"で説明するよ うに、マスキング機能を用いてブロックに対するアイコンおよびダイアログボッ クスをカスタマイズすることもできます。

モデルナビゲーションコマンド

サブシステムを使うと、多層階層のモデルを作成することができます。そして、 Simulink Model Browser(4-95 ページの"モデルの検索とブラウズ"を参照)か、あ るいはつぎのモデルナビゲーションコマンドを使用して、階層をナビゲーション することができます。

• Open

Open コマンドは、現在選択されているサブシステムを開きます。コマンドを実 行するには、Simulinkの Edit メニューから Open を選択し、Enter を入力する か、サブシステムをダブルクリックします。

• Go to Parent

Go to Parent コマンドは、カレントウィンドウに表示されているサブシステム の親を表示します。コマンドを実行するには、Esc を入力するか、Simulinkの View メニューから Go to Parent を選択してください。

ウィンドウの再利用

モデルナビゲーションコマンドがサブシステムや親を表示するウィンドウをカレ ントのウィンドウにするか、あるいは、新しいウィンドウにするかを指定できま す。ウィンドウを再利用すると、画面がウィンドウで乱雑になることが避けられ ます。それぞれのサブシステムのウィンドウを作成すると、サブシステムを親や 兄弟と並べて表示することができます。ウィンドウの再利用を設定するには、 Simulink の File メニューから Preferences を選択し、Simulink Preferences ダイアロ グボックスにリストされる Window reuse type オプションの中から1つを選択してください。

再利用タ イプ	Open 動作	Go to Parent (Esc) 動作
none	新しいウィンドウにサブシス テムが表示されます。	親ウィンドウが前面に移動し ます。
reuse	サブシステムがカレントウィ ンドウの親と置換えられます。	親ウィンドウがカレントウィ ンドウのサブシステムと置換 えられます。
replace	サブシステムは新しいウィン ドウに表示されます。	親ウィンドウを表示します。 サブシステムウィンドウは消 えます。
mixed	サブシステムはそれ自身の ウィンドウに表示されます。	親ウィンドウが前面にきます。 サブシステムウィンドウは消 えます。

サブシステム端子のラベリング

Simulink は、Subsystem ブロック上の端子にラベルを付けます。ラベルは、これ らの端子を通じてサブシステムをサブシステムの外部のブロックと接続する Inport ブロックと Outport ブロックの名前です。

端子のラベルは、つぎの手順で非表示(表示)にすることができます。

- サブシステムブロックを選択し、Format メニューから Hide Port Labels (あるいは Show Port Labels)を選びます。
- サブシステム内の Inport あるいは Outport ブロックを選び、Format メニューから Hide Name (あるいは Show Name)を選びます。
- ・ サブシステムブロックのパラメータダイアログの Show port labels を選択します。

つぎの図は、2 つのモデルを示しています。左側のサブシステムには 2 つの Inport ブロックと 1 つの Outport ブロックがあります。右側の Subsystem ブロッ クは、ラベルを付けた端子を示しています。



サブシステムへのアクセス制御

ライブラリの中にあるサブシステムへのユーザアクセス権を制御することができ ます。特に、モデル内のサブシステムの使用を許可しながら、ユーザがライブラ リサブシステムの内容を表示したり編集することができないようにすることがで きます。

ライブラリサブシステムへのアクセス権を制御するには、サブシステムのパラ メータダイアログボックスを開き、Access パラメータを ReadOnly や NoReadOrWrite に設定してください。最初のオプションは、ユーザがライブラリサ ブシステムの内容を見たり、ローカルにコピーすることはできますが、オリジナ ルのライブラリを修正することはできません。2番目のオプションは、ユーザ が、ライブラリサブシステムの内容を表示したりローカルコピーを作成したり パーミションを変更することができません。サブシステムのアクセスオプション に関するより詳しい説明は、9-228ページの Subsystem を参照してください。どち らのオプションもリンクを作成することで、モデル内のライブラリシステムを使 用することはできます (4-77ページの"ライブラリ"を参照してください)。

コールバックルーチンの使用法

ブロック線図またはブロックを、特定の方法で動作させた場合に実行される MATLAB表現を定義することができます。*コールバックルーチンと*呼ばれるこ れらの表現は、ブロックまたはモデルパラメータに関連付けられます。たとえ ば、ブロックの OpenFcn パラメータに関連付けられたコールバックは、そのブ ロック名をダブルクリックするかパスを変更したときに実行されます。

コールバックルーチンを定義し、それらをパラメータに関連付けるためには、 set_param コマンドを使用します(10-27 ページの set_param をご参照ください)。

たとえば、つぎのコマンドは、ユーザが mymodel の Test ブロックをダブルク リックすると、変数 testvar を評価します。

set_param('mymodel/Test', 'OpenFcn', testvar)

多くのモデルコールバックと関連したルーチンに対して、clutch システム (clutch.mdl)を調べてみることができます。

コールバックのトレーシング

コールバックのトレーシングによって、モデルを開いたりシミュレーションする ときに、Simulink がどのコールバックをどんな順番で呼び出すかを決めることが できます。コールバックトレーシングを使用可能にするには、Simulink Preferences ダイアログボックス (2-15 ページの "Simulink の設定"を参照)の Callback tracing オプションを選択するか、set_param(0, 'CallbackTracing', 'on')を実行し てください。このオプションによって、Simulink は呼び出すコールバックを MATLAB コマンドウィンドウにリスト表示します。

モデルコールバックパラメータ

つぎの表は、コールバックルーチンを定義することができるパラメータの一覧を 示し、それらのコールバックルーチンがいつ実行されるかを示します。動作が起

こる前か後に実行されるルーチンは、それぞれ、動作の直前か直後に発生します。

パラメータ	実行時
CloseFcn	ブロック線図を閉じる前
PostLoadFcn	モデルをロードした後。このパラメータに対する コールバックルーチンの定義は、モデルが既にロー ド済みであることを必要とするインタフェースを生 成するのに有効です。
InitFcn	モデルシミュレーションの開始時に呼び出し
PostSaveFcn	モデルを保存した後
PreLoadFcn	モデルをロードする前。このパラメータに対する コールバックルーチンの定義は、モデルが使用する 変数をロードするのに有効です。
PreSaveFcn	モデルを保存する前
StartFcn	シミュレーションを開始する前
StopFcn	シミュレーションを停止した後。出力は、StopFcn 実 行前にワークスペース変数やファイルに書き込まれ ます。

ブロックコールバックパラメータ

つぎの表は、ブロックコールバックルーチンを定義することができるパラメータの一覧を示し、それらのコールバックルーチンがいつ実行されるかを示します。
動作が起こる前か後に実行されるルーチンは、それぞれ、動作の直前か直後に発 生します。

パラメータ	実行時
CloseFcn	close_system コマンドを使用してブロックを閉じると き
CopyFcn	ブロックをコピーした後。コールバックは、 Subsystem ブロックに対して再帰的です(つまり、 CopyFcn パラメータが定義されているブロックを含 む Subsystem ブロックをコピーすると、そのルーチ ンも実行されます)。ルーチンは、add_block コマン ドを用いてブロックをコピーする場合も実行されま す。
DeleteFcn	ブロックを削除する前。このコールバックは、 Subsystem ブロックに対して再帰的です。
DestroyFcn	ブロックが削除される時
InitFcn	ブロック線図がコンパイルされる前とブロックパラ メータが評価される前。
LoadFcn	ブロック線図がロードされた後。このコールバック は、Subsystem ブロックに対して再帰的です。
ModelCloseFcn	ブロック線図を閉じる前。このコールバックは Subsystem ブロックに対して再帰的です。
MoveFcn	ブロックが移動またはサイズが変更されたとき。
NameChangeFcn	ブロック名またはパスを変更した後。Subsystem ブ ロックのパスを変更すると、それ自身の NameChangeFcn ルーチンを呼び出した後、それが含 むすべてのブロックに対してこの関数を再帰的に呼 び出します。

パラメータ	実行時
OpenFcn	ブロックを開くとき。このパラメータは、一般に Subsystem ブロックで使用します。ルーチンはブロッ クをダブルクリックするか、ブロックを引数として open_system コマンドを呼び出すと実行されます。 OpenFcn パラメータは、ブロックのオープンと関連 した通常の挙動、すなわちブロックのダイアログ ボックスを表示するかサブシステムをオープンする 動作を変更します。
ParentCloseFcn	ブロックを含むサブシステムを閉じる前に、または new_system コマンド (10-22 ページの new_system を参 照)により新しいサブシステムの一部とするとき。
PreSaveFcn	ブロック線図の保存前。このコールバックは、 Subsystem ブロックに対して再帰的です。
PostSaveFcn	ブロック線図の保存後。このコールバックは、 Subsystem ブロックに対して再帰的です。
StartFcn	ブロック線図がコンパイルされた後とシミュレー ションを開始する前。S-Function ブロックの場合は、 StartFcn は、ブロックの mdlProcessParameters 関数の 最初の実行のすぐ前に実行されます。詳細は、 Writing S-Functions の第3章の "Overview of the C MEX S-Function Routines" を参照してください。
StopFcn	シミュレーションの終了時。S-Function ブロックの場 合、StopFcn はブロックの mdlTerminate 関数の実行後 に実行されます。詳細は、Writing S-Functions の第 3 章の "Overview of the C MEX S-Function Routines" を参 照してください。
UndoDeleteFcn	ブロックの削除が元に戻されたとき。

パラメータ	実行時
CloseFcn	close_system コマンドを使用してブロックを閉じると き
CopyFcn	ブロックをコピーした後。コールバックは、 Subsystem ブロックに対して再帰的です(つまり、 CopyFcn パラメータが定義されているブロックを含 む Subsystem ブロックをコピーすると、そのルーチ ンも実行されます)。ルーチンは、add_block コマン ドを用いてブロックをコピーする場合も実行されま す。
DeleteFcn	ブロックを削除する前。このコールバックは、 Subsystem ブロックに対して再帰的です。
DestroyFcn	ブロックが削除される時
InitFcn	ブロック線図がコンパイルされる前とブロックパラ メータが評価される前。
LoadFcn	ブロック線図がロードされた後。このコールバック は、Subsystem ブロックに対して再帰的です。
ModelCloseFcn	ブロック線図を閉じる前。このコールバックは Subsystem ブロックに対して再帰的です。
MoveFcn	ブロックが移動またはサイズが変更されたとき。
NameChangeFcn	ブロック名またはパスを変更した後。Subsystem ブ ロックのパスを変更すると、それ自身の NameChangeFcn ルーチンを呼び出した後、それが含 むすべてのブロックに対してこの関数を再帰的に呼 び出します。

モデル作成のヒント

ここでは、モデル作成に役立つヒントを紹介します。

• メモリに関して

一般に、メモリが多いほど Simulink の性能はよくなります。

階層の使用

サブシステムの階層をモデルに追加すると、多くの場合、複雑なモデルほど その恩恵が得られます。ブロックをグループ化すると、最上位のモデルが単 純化され、モデルが読みやすく理解しやすくなります。詳細は、4-66ページ の"サブシステムの作成"を参照してください。モデルブラウザ(4-100ページの "モデルブラウザ"を参照してください)は、複雑なモデルについての有効な情 報を提供します。

• モデルのクリーニング

組織化されドキュメントが付けられたモデルは、読みやすく理解しやすいものです。信号ラベルとモデルの注釈は、モデルで起こっていることを説明するのに役立ちます。詳細は、4-39ページの"信号の名前"と4-24ページの"ブロック間でのラインの描画"を参照してください。

モデル化戦略

モデルのいくつかが同じブロックを使用する傾向がある場合、モデル内のこ れらのブロックの保存を簡単にできます。その場合、新規モデルを作成する ときに、このモデルを開くだけで、共通して使用するブロックをそこからコ ピーすることができます。ブロックの集合をシステム内に配置し、そのシス テムを保存することによって、ブロックライブラリを作成することができま す。そうすると、その名前を MATLAB コマンドウィンドウに入力することに よって、システムにアクセスすることができます。

一般にモデルを作成する場合、まず紙面上でそれを設計した後で、コン ピュータを用いてそれを作成します。それから、ブロックを1つのモデルに まとめる場合、ブロックを結線するラインを追加する前にブロックをモデル ウィンドウに追加します。このようにすると、ブロックライブラリを開く頻 度を少なくすることができます。

ライブラリ

この機能により、外部のライブラリからモデルにブロックをコピーし、ソースブ ロックを変更すると、コピーされたブロックを自動的にアップデートすることが できます。ライブラリを使うと、独自のブロックライブラリを開発するユーザ、 あるいは(ブロックセットのような)他から提供されるそれらのブロックを利用 するユーザには、モデル内に最新バージョンのブロックが含まれることが保証さ れます。

専門用語

この機能に関して使われる専門用語を理解することは、重要なことです。

Library – ライブラリブロックの集まり。File メニューから New Library を使って明示的に作成しなければなりません。

Library block — ライブラリ内のブロック。

Reference block – ライブラリブロックのコピー。

Link – 参照ブロックとそのライブラリブロックとの関係。ライブラリブロックを 変更したときに、Simulink は参照ブロックをアップデートします。

Copy-ライブラリブロックまたは別の参照ブロックからブロックを作成する操作

これらの専門用語を図に示します。



Library (Source)

Model or Library (Destination)

ライブラリの作成

ライブラリを作成するために、File メニューの New サブメニューから Library を 選択してください。Simulink は、Library: untitled というラベルの付いた新たな ウィンドウを表示します。タイトルのないウィンドウがすでに表示されている場 合は連続した番号が付けられます。 つぎのコマンドを使うと、コマンドラインからライブラリを作成することができます。

new_system('newlib', 'Library')

このコマンドは、'newlib'という名前の新たなライブラリを作成します。ライブ ラリを表示するためには、open_system コマンドを使ってください。これらのコ マンドは、第10章の"モデル構築コマンド"で説明します。

ライブラリからブロックをコピーする前にライブラリ名を付けなければ(保存しなければ)なりません。

ライブラリの修正

ライブラリを開くと、ライブラリは自動的にロックされ、内容を修正することが できません。ライブラリのロックを解除するためには、Edit メニューから Unlock Library を選択してください。ライブラリウィンドウを閉じると、ライブ ラリはロックされます。

ライブラリリンクの作成

ライブラリウィンドウからモデルウィンドウヘブロックのアイコンをコピーする ことによって、モデル内のライブラリブロックヘリンクすることができます (4-11 ページの"ウィンドウ間でのブロックのコピーと移動"を参照)。または、 ブロックをライブラリブラウザからモデルウィンドウにドラッグする(4-83 ペー ジの"ブロックライブラリの参照"を参照)ことによっても可能です。

モデルや、または別のライブラリにライブラリブロックをコピーする場合、 Simulink はライブラリブロックにリンクを作成します。参照ブロックは、ライブ ラリプロックのコピーです。ユーザは、参照ブロック内のブロックパラメータを 変更することはできますが、ブロックをマスクしたり、マスクされているブロッ クに対してはそれを編集することはできません。また、参照ブロックに対して コールバックパラメータを設定することもできません。サブシステムへのリンク の場合、参照サブシステムの内容を変更することができます。(4-79 ページの"リ ンクしているサブシステムの変更"を参照してください。)

ライブラリと参照ブロックは、*名前によって*リンクされます。つまり、参照ブ ロックは、コピーが作られたときに実際にその名前をもつ特定のブロックとライ プラリにリンクされます。

参照ブロックを更新しようとしたときに、MATLAB パス上で Simulink がライブ ラリブロックもソースライブラリも見つけることができない場合、リンクは、 *unresolved* 状態です。Simulink はエラーメッセージを出力し、赤の点線を使って

これらのブロックを表示します。エラーメッセージは、つぎのように表示されます。

Failed to find block "source-block-name" in library "source-library-name" referenced by block "reference-block-path".

状態の参照ブロックは(赤で)つぎのように表示されます。

 1	Bad Link	}
ļ	·	

Reference Block Name

誤ったリンクを修正するために、つぎのいずれかを行って下さい。

- リンクされていない参照ブロックを削除し、ライブラリブロックをユーザの モデル内にコピーして戻してください。
- 必要なライブラリを含むディレクトリを MATLAB パスに追加し、Edit メニューから Update Diagram を選択してください。
- 参照ブロックをダブルクリックしてください。現れたダイアログボックス上で、パス名を訂正し、Apply または Close をクリックしてください。

ライブラリリンクを無効にする

モデル内のリンクされているブロックを無効にすることができます。Simulink は モデルをシミュレーションするときに、無効にしたリンクを無視します。リンク を無効にするには、リンクを選び、モデルウィンドウの Edit かコンテキストメ ニューから Link options を選び、つぎに、Disable link を選択します。無効にした リンクを元に戻すには、Link Options メニューから Restore link を選択してくださ い。

リンクしているサブシステムの変更

Simulink では、ライブラリリンクのサブシステムを変更することができます。サ ブシステムの構造を変えるような構造的な変更の場合、参照ブロックからライブ ラリブロックへのリンクを解除しなければなりません。サブシステムのリンク構 造を変更しようとすると、Simulink はリンクを解除するように促します。構造的 な変更とは、ブロックやラインを加えたり削除したり、あるいは、ブロックの端 子の数を変えたりするような変更のことです。非構造的な変更とは、サブシステムの構造に影響を与えないようなパラメータ値の変更のことです。

リンクサブシステムの変更をライブラリに反映させる

Simulink では、構造的な変更ではなく、非構造的な変更のあるアクティブなリン クをもつことができます。構造的な変更があるリンクを再び保存しようとする と、Simulink は、変更をオリジナルのリンク元のライブラリに反映させるか、そ れとも破棄するかを尋ねます。ここで、変更を反映させるを選ぶと、Simulink は、変更された参照ブロックでライブラリブロックを更新します。どちらにして も、最終的に参照ブロックはライブラリブロックをそのままコピーしたものにな ります。

非構造的な変更のあるリンクを元に戻す場合は、Simulink は変更を反映させるか 破棄するかを尋ねることなしに、リンクを利用可能にします。後で、変更を反映 させたり破棄したい時は、参照ブロックを選択し、モデルウィンドウの Edit メ ニューかコンテキストメニューより Link options を選択し、Propagate/Discard changes を選んでください。参照ブロックと対応するライブラリブロックの非構 造的パラメータの違いを表示するには、Link options メニューから View changes を選んでください。

リンクされたブロックのアップデート

Simulink は、以下の場合に、モデルまたはライブラリ内の参照ブロックをアップ デートします。

- モデルまたはライブラリがロードされたとき
- ・ Edit メニューから Update Diagram を選択するか、シミュレーションを実行した とき
- get_param コマンド(4-81ページの"ライブラリのリンク状態"を参照)を使って、 ブロックの LinkStatus パラメータを参照したとき
- find_system コマンドを使ったとき

ライブラリブロックに対するリンクの解除

参照ブロックを、ライブラリブロックにリンクされていない、ライブラリブロッ クの単なるコピーにするために、参照ブロックとそのライブラリブロック間のリ ンクを解除することができます。ライブラリブロックに対する変更は、ブロック に影響を与えなくなります。ライブラリブロックに対するリンクを解除すると、 ライブラリ無しのスタンドアロンモデルとして、モデルを変換することができます。

参照ブロックとライブラリブロック間のリンクを解除するためには、参照ブロッ クを選択し、そして、Edit メニューから Break Library Link を選びます。つぎのコ マンドを使って、LinkStatus パラメータの値を 'none' に変更することで、コマン ドラインから参照ブロックとライブラリブロック間のリンクを解除することもで きます。

set_param('refblock', 'LinkStatus', 'none')

つぎのコマンドを使って、システムを保存し、参照ブロックとライブラリブロック間のすべてのリンクを解除することができます。

save_system('sys', 'newname', 'BreakLinks')

参照ブロックに対するライブラリブロックの検出

参照ブロックにリンクされたソースライブラリとブロックを検出するためには、 参照ブロックを選択し、モデルウィンドウの Edit メニューかあるいはコンテキ ストメニューの Link options サブメニューから Go To Library Link を選択してく ださい。ライブラリが開いていれば、Simulink はライブラリブロックを選択し(ブロック上に選択状態を示すハンドルを表示)、ソースライブラリをアクティブ ウィンドウにします。ライブラリが開いてなければ、それを開いてライブラリブ ロックを選択します。

ライブラリのリンク状態

すべてのブロックが、そのブロックが参照ブロックであるかどうかを示す LinkStatus パラメータをもっています。パラメータはつぎの値をもちます。

状態	説明
none	ブロックが参照ブロックでないことを示します。
resolved	リンクが resolved 状態であることを示します。
unresolved	リンクが unresolved 状態であることを示します。
implicit	ブロックがリンクされたブロックの中にあることを示しま す。
inactive	リンクが無効な状態であることを示します。

ライブラリリンクの表示

オプションで、モデルにライブラリリンクが存在すると、それぞれのアイコンの 左下角に矢印を表示します。



この矢印によって、アイコンがライブラリブロックとリンクしているか、あるい はローカルなブロックかを一目で判断できます。ライブラリリンクを表示するに は、モデルウィンドウの Format メニューから Library Link Display を選択し、つ ぎに User(ユーザライブラリへのリンクのみ表示)あるいは、All(すべてのリン クを表示)のどちらかを選択してください。

リンク矢印の色は、リンクの状態を示しています。

色	状態	
黒	アクティブなリンク	
グレイ	アクティブでないリンク	
赤	アクティブで変更されているリンク	

ライブラリブロックについての情報の取得

システム内の参照ブロックについての情報を得るために、libinfo コマンドを使い ます。コマンドの形式は、つぎの通りです。

libdata = libinfo(sys)

ここで、sys はシステム名です。このコマンドは、サイズが n × 1 の構造体を出 力します。n は、sys 内のライブラリブロックの数です。構造体の各要素は、以 下の 4 つのフィールドをもちます。

• Block: ブロックのパス

- Library: ライブラリ名
- ReferenceBlock:参照ブロックのパス
- LinkStatus:リンク状態、'resolved' または 'unresolved'

ブロックライブラリの参照

Library Browser を使用すると、即座にライブラリブロックを見つけ出し、モデル にコピーすることができます。Library Browser を表示するには、MATLAB デス クトップ、あるいは Simulink モデルウィンドウのツールバーから Library Browser ボタンをクリックするか、MATLAB コマンドラインに Simulink と入力し てください。

注意 ライブラリブラウザは、Microsoft Windows のみで利用可能です。



ライブラリブラウザは3つのパネルからなります。

ツリーパネルには、インストールされているすべてのブロックライブラリが表示 されます。アイコンパネルには、ツリーパネルの中で現在選択されているライブ ラリに属するブロックのアイコンが表示されます。ドキュメントパネルには、ア イコンパネルで選択されたブロックのドキュメントが表示されます。

ブロックは、ライブラリブラウザのライブラリツリーをナビゲーションするか、 ライブラリブラウザの検索機能を使用することで見つけ出すことができます。

ライブラリツリーのナビゲーション

ライブラリツリーには、システムにインストールされているすべてのブロックラ イブラリが表示されます。マウスやキーボードを使って、拡大、縮小して、ライ ブラリの内容を表示したり隠すことができます。ツリーを拡大/縮小するには、 ライブラリの横の +/- ボタンをクリックするかキーボードの +/- か右 / 左 矢印 キーを押します。ツリーを上下に移動するには、上 / 下 矢印キーを使用します。

ライブラリの検索

特定のブロックを見つけ出すために、ライブラリブラウザの Find ボタンのとな りの入力フィールドにブロック名を入力し、Find ボタンをクリックします。

ライブラリのオープン

ライブラリをオープンするために、ブラウザ内でライブラリ要素を右クリックし ます。Simulink は Open Library ボタンを表示します。ライブラリをオープンす るために、Open Library ボタンを選択します

モデルの作成とオープン

モデルを作成するために、ライブラリブラウザのツールバーで New ボタンを選 択します。既存のモデルをオープンするために、ツールバーで Open ボタンを選 択します。

ブロックのコピー

ライブラリブラウザからモデルにブロックをコピーするために、ブラウザでブ ロックを選択し、モデルウィンドウまで選択したブロックをドラッグし、コピー を作成したい場所にドロップします。

ブロックに関するヘルプの表示

ブロックに関するヘルプを表示するために、ブラウザでブロックを右クリックし、その後現れたボタンを選択します。

ライブラリブラウザを最前面にホールド

ライブラリブラウザをデスクトップ上のウィンドウの最前面に保持するために、 ブラウザのツールバーの **押しピン** ボタンを選択します。

ライブラリブラウザにライブラリを追加

作成したライブラリをライブラリブラウザに表示したい場合は、ライブラリを記述する slblocks.m ファイルをそのライブラリを含むディレクトリに作成する必要があります。slblocks.m ファイルを作成する最も簡単な方法は、既存の slblocks.m ファイルをテンプレートとして使うことです。MATLAB プロンプトで

which('slblocks.m', '-all')

とタイプすることにより、システム上ですべての既存の slblocks.m を見つけるこ とができます。表示されたファイルをライブラリのディレクトリにコピーしてく ださい。それから、コピーしたファイルをオープンし、ファイル内の指示に従っ て編集し、結果を保存します。最後に、必要ならばライブラリのディレクトリを MATLAB パスに追加してください。次回ライブラリブラウザをオープンしたと きに、ライブラリは他のライブラリと共にプラウザに表示されます。

方程式のモデル化

新しい Simulink のユーザをもっとも悩ます問題の1つは、方程式をどのように モデル化するかということです。ここでは、方程式をモデル化する方法について の理解を深めるために役立ついくつかの例を示します。

摂氏から華氏への変換

摂氏温度を華氏温度に変換する方程式をモデル化するには、

 $T_F = 9/5(T_C) + 32$

まずモデルを作成するのに必要なブロックを検討します。

- Sources ライブラリから、温度信号を入力する Ramp ブロック
- ・同じく Sources ライブラリから、定数 32 を定義する Constant ブロック
- Math ライブラリから、入力信号に 9/5 を乗算する Gain ブロック
- 同じく Math ライブラリから、2 つの量を加算する Sum ブロック
- Sinks ライブラリから、出力を表示する Scope ブロック

つぎに、ブロックをモデルウィンドウに集めます。



各ブロックを (ダブルクリックして)開き、適切な値を入力することによって、 Gain ブロックと Constant ブロックにパラメータ値を割り当てます。それから、 Close ボタンをクリックして値を適用し、ダイアログボックスを閉じます。

つぎに、ブロックを結線します。



Ramp ブロックには、摂氏温度を入力します。そのブロックを開き、Initial output パラメータを0に変更します。Gain ブロックはその温度に定数の 9/5 を乗 算します。Sum ブロックは、結果に値 32 を加えて華氏温度を出力します。

Scope ブロックをオープンし、出力を表示します。つぎに、Simulation メニュー から Start を選択してシミュレーションを実行します。シミュレーションは 10 秒間実行されます。

簡単な連続システムのモデル化

つぎの微分方程式をモデル化します。

x'(t) = -2x(t) + u(t)

ここで、u(t) は振幅 1 と周波数 1 ラジアン / 秒の矩形波です。Integrator ブロック は、その入力 x'を積分して x を求めます。このモデルで必要な他のブロックに は、Gain ブロックと Sum ブロックがあります。矩形波を生成するには、Signal Generator ブロックを用いて Wave form として Square を選択し、デフォルトの単 位を rad/sec に変更します。Scope ブロックを用いて再び出力を表示します。ブ ロックを集めて Gain を定義します。

このモデルでは、Gain ブロックの方向を反転するために、ブロックを選択した 後で Format メニューから Flip Block コマンドを使用します。また、Integrator ブ ロックの出力から Gain ブロックへの分岐線を生成するために、Ctrl キーを押し ながらラインを描きます。詳細は、4-25 ページの"分岐線の描画"を参照してくだ さい。これで、すべてのブロックを結線することができます。



このモデルの重要な概念は、Sum ブロック、Integrator ブロック、および Gain ブ ロックを含むループです。この方程式で、x は Integrator ブロックの出力です。x はまた、その基になる x ´を計算するブロックに対する入力でもあります。この 関係はループを用いて実現されます。 Scope は、各時間ステップごとに x を表示します。10 秒間のシミュレーションの 場合、出力はつぎのようになります。



この例でモデル化した方程式は、伝達関数として表現することもできます。モデ ルは Transfer Fcn ブロックを用いて、入力として u を受け入れ、x を出力します。 したがって、このブロックは x/u を実現します。上記の方程式で x ´に sx を代 入すると、つぎのようになります。

sx = -2x + u

xに対する解は

x = u/(s+2)

または、

x/u = 1/(s+2)

Transfer Fcn ブロックは、分子と分母の係数を指定するパラメータを使います。 この場合、分子は1で、分母は s+2 です。いずれの項も s のべき乗の降順に係数 のベクトルとして指定します。この場合、分子は[1](または単に1)で分母は[1 2]です。するとモデルはつぎのようにきわめて単純なものになります。



このシミュレーションの結果は、前のモデルの結果と同一です。

モデルの保存

モデルは、File メニューから Save または Save As コマンドを選択することによっ て保存することができます。Simulink では、ブロック線図とブロックプロパティ を含んだモデルファイル (.mdl という拡張子をもつ)と呼ばれる特別にフォー マットされたファイルを生成しモデルを保存します。モデルファイルのフォー マットについては、付録 B の"モデルファイルフォーマット"で説明します。

初めてモデルを保存する場合には、Save コマンドを用いてモデルファイルの名 前と保存場所を指定します。モデルファイル名は文字で始まる 31 字以下の文字、 数字、アンダースコアの組合わせでなければなりません。

以前に保存していたモデルファイルを保存する場合、Save コマンドを用いて ファイルの内容を置き換えるか、Save As コマンドを用いてモデルを新しい名前 で、あるいは新しい場所に保存します。

Simulink は、モデルを保存するまでにつぎの手順を実行します。

- モデルに対する mdl ファイルが既に存在する場合、テンポラリファイルとして別名に変更します。
- Simulink は全てのブロックの PreSaveFcn コールバックルーチンを実行し、それからブロック線図の PreSaveFcn コールバックルーチンを実行します。
- Simulink は、同じ名前と拡張子 mdl を使ってモデルファイルを新しいファイル に書き出します。
- 4 Simulink は全てのブロックの PostSaveFcn コールバックルーチンを実行し、それからブロック線図の PostSaveFcn コールバックルーチンを実行します。
- 5 Simulink は、テンポラリファイルを削除します。

この手順を実行中にエラーが発生すると、Simulink はテンポラリファイルをオリ ジナルモデルファイルの名前に変更し、現在のバージョンのモデルを拡張子.err のファイルに書き出し、エラーメッセージを発行します。Simulink は、エラーが 早い段階で発生しても、ステップ2からステップ4までは実行します。

ブロック線図の印刷

ブロック線図は、File メニューから Print を選択する (Microsoft Windows システムの場合)か、MATLAB コマンドウィンドウで print コマンドを用いて (すべての プラットフォームにおいて)印刷することができます。

Microsoft Windows システムの場合、Print メニュー項目はカレントウィンドウ内のブロック線図を印刷します。

Print ダイアログボックス

Print メニュー項目を選択すると、Print ダイアログボックスが現れます。Print ダイアログボックスにより、モデル内のシステムを選択して印刷することができ ます。ダイアログボックスを使って、以下のことが行えます。

- カレントシステムのみを印刷
- カレントシステムとモデル階層内の上層のすべてのシステムを印刷
- カレントシステムとモデル階層内の下層のすべてのシステムを、マスクブ ロックとライブラリブロックの内容を見るためのオプション付きで印刷
- モデル内のすべてのシステムを、マスクブロックとライブラリブロックの内容を見るためのオプション付きで印刷
- 各ダイアグラムにフレームをかぶせて印刷

選択印刷をサポートする Print ダイアログボックスの部分は、サポートされてい る各プラットフォームで同じものです。Microsoft Windows システム上でのダイ アログボックスの一部を以下に示します。この図では、カレントシステムのみの 印刷が選択されています。



Current system and below または All systems オプションのいずれかを選択すると、2 つのチェックボックスが使用不可能になります。つぎの図では、All systems が 選択されています。



Look Under Mask Dialog チェックボックスを選択すると、カレントブロックのレベルまたは下層レベルでマスクされたサブシステムの内容を印刷します。すべてのシステムを印刷する場合、最上位のシステムがカレントブロックと考えられるので、Simulink はマスクされたプロックのどの内容でも印刷します。

Expand Unique Library Links チェックボックスを選択すると、ブロックがシステムの場合、ライブラリブロックの内容を印刷します。モデル内にブロックのコピーがいくつ含まれていようと、1 つだけ印刷されます。ライブラリに関する詳細は、4-77 ページの"ライブラリ"を参照してください。

印刷ログは、印刷されたブロックとシステムをリストします。印刷ログを印刷するためには、Include Print Log チェックボックスを選択してください。

Frame チェックボックスを選択すると、各ダイアグラム上にタイトルブロック フレームを印刷します。隣接したエディットボックスにタイトルブロックフレー ムまでのパスを入力します。MATLABのフレームエディタを利用して、特殊な タイトルブロックフレームを作成することができます。タイトルブロックフレー ムを作成するためのフレームエディタの利用に関する詳細は、frameeditの項目 を参照してください。

Print コマンド

print コマンドの書式は、つぎの通りです。

print -- ssys -- device filename

sys は、印刷するシステムの名前です。システム名の前には s スイッチ識別子がなければならず、これは唯一必要な引数です。sys は、カレントセッション中に オープンするかあらかじめオープンされていなければなりません。システム名に スペースが含まれたり、複数行を必要とする場合には、名前を文字列として指定 する必要があります。下の例を参照してください。

device は、デバイスタイプを指定します。デバイスタイプの一覧と説明について は、Using MATLAB Graphics を参照してください。

filename は、出力を保存する PostScript ファイルです。filename が既に存在する場合、それは置き換えられます。filename に拡張子がない場合、適切な拡張子が付けられます。

たとえば、つぎのコマンドは untitled という名前のシステムを印刷します。

print -suntitled

つぎのコマンドは、カレントシステム内の Sub1 というサブシステムの内容を印 刷します。

print -sSub1

つぎのコマンドは、Requisite Friction という名前のサブシステムの内容を印刷します。

print (['-sRequisite Friction'])

つぎの例では、名前が2行におよぶサブシステム Friction Model を印刷します。 最初のコマンドはニューラインキャラクタを変数に割り当て、2行目でシステム を印刷します。

cr = sprintf('\n'); print (['-sFriction' cr 'Model'])

選択されているサブシステムを印刷するには、以下のように入力します。

print(['-s', gcb])

用紙の大きさと方向の指定

Simulink は、モデルダイアグラムを印刷するために利用する用紙の型と方向を指定することができます。set_param コマンドを使ってモデルの PaperType プロパティと PaperOrientation プロパティをそれぞれ設定することで、全てのプラットフォー

ムにおいて上記のことが実行できます (A-1 ページの"モデルとブロックパラ メータ"を参照してください)。MATLABの orient コマンドを使って、用紙の方 向のみを設定することができます。Windows では、Print ダイアログボックスお よび Printer Setup ダイアログボックスを使って、同様に用紙の型と方向のプロパ ティを設定することができます。

ダイアグラムの位置と大きさ

モデルの PaperPositionMode パラメータと PaperPosition パラメータを使って、印刷 ページ上でのモデルの位置と大きさを設定できます。PaperPosition パラメータ値 は、[left bottom width height] のベクトルです。はじめの2つの要素は、ページ上 での長方形領域の左下隅を指定します。最後の2つの要素は、長方形の幅と高さ を指定します。モデルの PaperPositionMode が manual のとき、Simulink は指定し た印刷長方形領域の内側にフィットするようにモデルダイアグラムを(必要であ ればスケーリングして)配置します。たとえば、つぎのコマンド

vdp

set_param('vdp', 'PaperType', 'usletter')
set_param('vdp', 'PaperOrientation', 'landscape')
set_param('vdp', 'PaperPositionMode', 'manual')
set_param('vdp', 'PaperPosition', [0.5 0.5 4 4])
print -svdp

は、サンプルモデル vdp のブロックダイアグラムを横向きに U.S. レターサイズ のページの左下隅に印刷します。

PaperPositionMode が auto のとき、Simulink はページにフィットするように必要 であればダイアグラムをスケーリングして、印刷ページ上の中心にモデルダイア グラムを配置します。

モデルの検索とブラウズ

Simulink は、モデルの検索とブラウズのためのツールを提供します。これらは、 オブジェクトを表示したり修正する必要があり、その位置がわからないときに役 立ちます。

オブジェクトの検索

モデル内のブロック、信号、状態、その他のオブジェクトを見つけ出すために、 Simulink の Edit メニューから Find を選択します。すると、Simulink は Find ダイ アログボックスを表示します。

-1	Find : vdp		_ 🗆 ×
	Filter options	_ Search criteria	
	Lookfar Dalad	Basic Advanced	Find
		Dasic Auvanceu	Heln
	🗄 🙀 Simulink objects 🔛	Find what:	
	🛨 🙀 Stateflow objects 🛛 🔽		Cancel
		🗖 Search block dialog parameters	
		🗖 Match case Contains word 🔻	
	1		
	🗖 Look inside masked systems	Start in system	
	Look inside linked systems	vdp 🔽	
Ŀ			

Find dialog box を使ってオブジェクトを検索するには、まず Filter options (4-97 ページの"フィルタオプション"を参照)および Search criteria (4-97 ページの"検 索基準"を参照)パネルを使って検索したいオブジェクトの特性を指定します。 つぎに、複数のシステムまたはサプシステムがオープンされている場合は、 Finder の Start in system リストから検索を開始したいシステムまたはサプシステ ムを選択します。最後に、Find ボタンを選択します。Finder は、指定した条件 を満たすオブジェクトについて選択したシステムを検索します。条件を満たすオ ブジェクトは、Finder ダイアログボックスの一番下の結果パネルに表示されま す。

🕽 Find : vdp				
Filter options Look for Select Simulink objects Select Stateflow objects Select Look inside masked systems Look inside linked systems	Search criteria Basic Advanced Find what: Mul Search block dialog parameters Match case Contains word Start in system Vdp	Find Help Cancel		
Type Name	Parent 8	Source Destination		
Slock Mu	vdp			
Block Mux	vdp			
Found 2 object(c)				

検索結果リストの項目をダブルクリックしてオブジェクトを表示することができ ます。Simulink は、(必要ならば)オブジェクトを含むシステムまたはサブシス テムをオープンし、オブジェクトを強調して選択します。結果リストを並び替え るには、各列の一番上のボタンをクリックしてください。たとえば、オブジェク トのタイプ順に結果を並び替えるには、Type ボタンをクリックします。ボタン を1回クリックすると、昇順にリストを並び替え、2回クリックすると降順に並 び替えます。オブジェクトのパラメータまたはプロパティを表示するには、リス ト内のオブジェクトを選択します。その後、右マウスボタンを押して結果のコン テキストメニューから Parameter または Properties を選択します。

フィルタオプション

Filter options パネルを使って、検索するオブジェクトの種類や検索する場所を指 定することができます。



Object type list. object type list は、Finder が検索するオブジェクトのタイプの一覧 を表示します。タイプのチェックを外すと、Finder の検索から除外することがで きます。

Look inside masked subsystem. このオプションをチェックすると、Finder は、マス クされたサブシステム内のオブジェクトを検索します。

Look inside linked systems. このオプションをチェックすると、Finder はライブラリ にリンクされたサブシステム内のオブジェクトを検索します。

検索基準

Search criteria パネルを使って、ユーザの検索条件を満足するためにオブジェクトが満たすべき条件を指定することができます。

Basic. Basic パネルを使って、名前とオプションでダイアログパラメータが指定 したテキスト文字列と一致するオブジェクトを検索します。パネルの Find what フィールドに検索テキストを入力します。前の検索テキストを表示するには、 Find what フィールドのとなりのドロップダウンリストボタンを選択してくださ い。テキストを再入力するためには、ドロップダウンリスト内でクリックしてく ださい。検索にダイアログパラメータを含ませたい場合は、Search block dialog parameters をチェックしてください。

Advanced. Advanced パネルを使って、オブジェクトが検索条件を満足するべきプロパティを7個指定できます。

Basic	Advanced		
Selec	t Property		Value
	(none)	▼	
1			

プロパティを指定するには、Advanced パネルの Property 列のセルのうちの1つ に名前をタイプするか、セルのドロップダウンプロパティリストからプロパティ を選択します。リストを表示するには、セルの隣の下向き矢印ボタンを選択しま す。つぎに、プロパティ名の隣の Value 列にプロパティ値を入力します。プロパ ティ名を入力するとき、Finder は、Select 列のプロパティ名の隣のチェックボッ クスをチェックします。これは、プロパティが検索に含まれることを示します。 プロパティを除外したい場合は、チェックボックスのチェックを外します。

Match case. オブジェクトプロパティの値に対して検索テキストのマッチングを 行うときに、大文字と小文字の区別を考慮に入れる場合にこのオプションを チェックします。

Other match options. Match case オプションの隣は、選択が可能な他のマッチング オプションを指定するドロップダウンリストです。

· Match whole word

プロパティ値と検索テキストが大文字か小文字かを除いて同一である場合に マッチングを指定

· Contains word

プロパティ値が検索テキストを含む場合にマッチングを指定

• Regular expression

検索テキストがプロパティ値に対してマッチングを行うときに正規表現とし て取り扱われることを指定します。つぎのキャラクタは、正規表現中では特 殊な意味をもちます。

キャラクタ	意味
٨	文字列の先頭のマッチング
\$	文字列の末尾のマッチング
	任意のキャラクタのマッチング
١	エスケープキャラクタ。つぎのキャラクタは通常の意味をも ちます。たとえば、正規表現 \ は、.a および .2 とピリオドで 始まる wo- キャラクタ文字列がマッチします。
*	その前の0個またはそれ以上のキャラクタがマッチ。たとえ ば、ba*は、b,ba,baa 等がマッチします。
+	その前の1個またはそれ以上のキャラクタがマッチ。たとえ ば、ba+ は、ba, baa 等がマッチします。
0	カレントキャラクタにマッチするキャラクタを示す。ハイフ ンはキャラクタの範囲を示すために用います。たとえば、 [a-zA-Z0-9_]+は、foo_bar1とマッチしますが、foo\$barとは マッチしません。^は、カレントキャラクタがつぎのキャラ クタのうちの1つでないとき、マッチすることを示します。 たとえば、[^0-9]は、数字でない任意のキャラクタがマッチ します。
\mathbf{w}	ワードキャラクタのマッチング ([a-z_A-Z0-9]) と同じ
$\setminus W$	非ワードキャラクタのマッチング ([^a-z_A-Z0-9]) と同じ
\d	数字のマッチング ([0-9] と同じ)
\D	非数字のマッチング ([^0-9] と同じ)
\s	空白のマッチング ([\t\r\n\f] と同じ)

キャラクタ	意味
$\setminus S$	非空白のマッチング ([^ \t/r/n/f] と同じ)
\ <word\></word\>	WORD のマッチング。WORD は空白で囲まれたワードキャ ラクタからなる文字列です。

モデルブラウザ

モデルブラウザを使って、以下のことが行えます。

- モデルの階層構造を調べる。
- モデル内のシステムを直接オープンする。
- モデル内に含まれるブロックを決定する。
- source control system を使ってモデルを管理する。"MATLAB User's Guide の第8章の "Interfacing with Source Control Systems" を参照してください。

ブラウザの操作は、Microsoft Windows と UNIX とで異なります。

Windows でのモデルブラウザの使用

Model Browser パネルを表示するには、Simulink **View** メニューから **Model Browser** を選択します。モデルウィンドウは、2 つのパネルに分割されます。左 のパネルは、右側に表示されたブロック線図をツリー構造で表示します。

注意 Browser initially visible プリファレンスにより、デフォルトでモデルブラウ ザ上にモデルがオープンされます。このプリファレンスを設定するには、 Simulink も File メニューから Preferences を選択してください。



ツリー構造一覧での各要素は、モデル内のサブシステムに対応します。各サブシ ステムの横の +/- ボックスをクリックすることで、または左 / 右矢印キーかテン キー上の +/- キーを押すことで、ツリー構造を拡大 / 縮小することができます。 キーボードの矢印キーを押すことで、ツリー構造を上 / 下に移動できます。ダイ アグラムビューでサブシステムの内容を表示するためには、サブシステムをク リックします。サブシステムに関連する新しいウィンドウをオープンするために は、ダイアグラムビューでサブシステムをダブルクリックします。また、マウス とキーボードを使って、ツリー構造表示のサブシステムを素早くナビゲーション することができます。

マウスでナビゲーション.サブシステムを選択するために、ツリー構造からサブシ ステムをクリックします。サブシステムに含まれているサブシステムをリストす るには、サブシステムの横の+ボタンをクリックします。サブシステムを縮小 するには、もう一度ボタンをクリックします。

キーボードでナビゲーション.ツリー構造を上下に移動するには、上/下矢印キー を使用します。テンキーの左/右矢印や+/-キーを使用すると、サブシステムを 含む全体が拡大されます。 **ライブラリリンクの表示**. モデルブラウザは、モデルのツリー構造からライブラリ リンクを取り込んだり除外することができます。デフォルトでは、ライブラリリ ンクを表示するかどうかを指定するために、Simulink Preferences ダイアログボッ クスを使用します。ライブラリリンクの表示を切り替えるには、Simulinkの View メニューの Model browser options サブメニューから Show library links を選択 します。

マスクされたサプシステムの表示. モデルブラウザは、ツリー構造一覧にマスクさ れたサブシステムを含めたり、除外したりすることができます。ツリー構造にマ スクされたサブシステムを含める場合は、ツリー構造のマスクされたサブシステ ムを選択すると、ダイアグラム表示にブロック線図を表示します。デフォルト で、マスクされたサブシステムを表示するかどうかを指定するには、Simulink Preferences ダイアログボックスを使用します。マスクされたサブシステムの表示 の切り替えは、Simulinkの View メニューの Model browser options サブメニュー から Look under masks を選択してください。

UNIX でのモデルブラウザの使用

モデルブラウザパネルを表示するには、File メニューから Show Browser を選択 します。モデルブラウザが表示され、カレントモデルの情報が表示されます。こ の図は、クラッチシステムの内容を表示したモデルブラウザウィンドウを示して います。



モデルウィンドウの内容

モデルブラウザウィンドウは、以下のもので構成されます。

- システムのリスト。左側のリストは、選択されたカレントシステムについて、 カレントシステムとそれに含まれるサブシステムを含んでいます。
- ブロックのリスト。右側のリストは、選択されたシステム内のブロックの名前を含んでいます。初期状態では、このウィンドウは最上位システム内のブロックを表示します。
- File メニュー。Print, Close Model, Close Browser メニュー項目を含んでいます。
- Options メニュー。Open System, Look Into System, Display Alphabetical/ Hierarchical List, Expand All, Look Under Mask Dialog, Expand Library Links メ ニュー項目を含んでいます。
- Optionsチェックボックスとボタン。Look Under [M]ask DialogとExpand [L]ibrary Links チェックボックス、Open System と Look Into System ボタン。デフォルト で、Simulink は、マスクされたブロックとライブラリリンクのブロックの内容 を表示しません。これらのチェックボックスにより、デフォルトを変更する ことができます。
- 選択したブロックのブロックタイプ
- ダイアログボックスのボタン。Help, Print, Close ボタンがあります。

リスト内容の解釈

Simulink は、マスクされたブロック、参照ブロック、OpenFcn パラメータが設定 されたブロック、サブシステムを含むシステムを、ブロック名やシステム名の前 に以下の記号を使って識別します。

- システムリスト内のシステム名の前にプラス記号(+)がある場合、システムが 拡張可能であることを示します。これは、下層システムをもっていることを 意味します。システム名をダブルクリックすると、リストが拡張され、ブ ロックリストにその内容が表示されます。システムが拡張されると、その名 前の前にはマイナス記号(-)が現れます。
- [M]は、ブロックがマスクされていることを示します。ブロックは、マスクダ イアログボックスまたはマスクワークスペースのいずれかをもちます。マス キングに関する詳細は、第7章の"マスクを使ったブロックのカスタマイズ"を 参照してください。
- [L] は、ブロックが参照ブロックであることを示します。詳細は、4-24 ページの "ブロックの接続"を参照してください。
- [O] は、ブロックに対して open function (OpenFcn) コールバックが定義されていることを示します。ブロックのコールバックの詳細は、4-71 ページの"コールバックルーチンの使用法"を参照してください。

• [S] は、システムが Stateflow ブロックであることを示します。

システムをオープンする

ブロックリストに現れている名前をもつブロックまたはシステムをオープンする ことができます。システムをオープンするには、つぎのようにします。

- 1 システムのリストで、オープンしたいシステムを含む親システムの名前をシ ングルクリックにより選択してください。親システムの内容は、ブロックの リストに現れます。
- 2 システムがマスクされているか、ライブラリブロックによってリンクされているか、あるいはオープンファンクションコールバックをもっているかによって、つぎのようにしてオープンします。
 - システムの左側に記号がなければ、その名前をダブルクリックするか、その名前を選択して、Open System ボタンをクリックしてください。
 - システムの名前の前に [M] または [O] があれば、システム名を選択して、 Look Into System ボタンをクリックしてください。

マスクされたシステムまたはリンクされたブロックの参照

デフォルトで、モデルブラウザは、([M] で識別される) マスクされたシステムと ([L] で識別される) リンクされたブロックをブロックではあってもサブシステム ではないと考えます。マスクされたシステムまたはリンクされたブロックが選択 されているときに Open System をクリックすると、モデルブラウザはシステムま たはブロックのダイアログボックスを表示します (Open System は、ブロック線 図内のブロックをダブルクリックするのと同様に機能します)。同様に、ブロッ クの OpenFcn コールバックパラメータが定義されている場合は、ブロックが選択 されている間に Open System をクリックするとコールバック関数が実行されま す。

ブロックリスト内のブロックを選択し、Look Into System をクリックすること で、ダイアログボックスまたはコールバック関数機能を参照するために、モデル ブラウザを利用することができます。モデルブラウザは、基礎となるシステムま たはブロックを表示します。

リスト内容のアルファベット順の表示

デフォルトで、システムのリストは、モデルの階層を示します。システムを含む システムには、プラス記号 (+) がその前に表示されます。これらのシステムが拡 張されると、モデルブラウザは、それらの名前の前にマイナス記号 (-) を表示し ます。システムをアルファベット順に表示するためには、Options メニューの Display Alphabetical List メニュー項目を選択してください。

モデルのバージョン管理

Simulink は、モデルの複数のバージョンを管理するために役立つ機能を提供します。

- モデルを編集すると、Simulink はバージョン番号、モデルの作成者と変更者、 オプションの変更履歴を含むモデルに関するバージョン管理情報を生成しま す。Simulink は、自動生成したモデルに関するバージョン管理情報を保存しま す。詳細は、4-114ページの"バージョン管理プロパティ"を参照してください。
- Simulink の Model Parameters ダイアログボックスを利用して、モデルに記録されたバージョン管理情報を編集し、バージョン管理のオプションを選択することができます(4-109 ページの"モデルプロパティダイアログ"を参照)。
- Simulink Model Info ブロックを利用して、外部のバージョン管理システムで保存されている情報も含め、モデルダイアグラムの注釈ブロックとしてバージョン管理情報を表示することができます(9-159ページの"Model Info"を参照)。
- Simulink バージョン管理パラメータを利用して、MATLAB コマンドラインや M-ファイルからバージョン管理情報にアクセスすることができます。
- Simulinkの File メニューから Source Control サブメニューを使用して、ユーザ ソースコントロールシステムにモデルをチェックイン、チェックアウトさせ ることができます。より詳細な情報については、MATLAB ドキュメントの "Interfacing with Source Control Systems," を参照してください。

カレントユーザの指定

モデルを作成したり更新したとき、Simulink はバージョン管理の目的でモデルに ユーザ名を記録します。Simulink は、つぎの環境変数 USER, USERNAME, LOGIN, LOGNAME の内、少なくとも一つを利用してユーザ名が指定されている ことを仮定します。これらの変数のどれも定義されていないとき、Simulink はモ デルのユーザ名を更新しません。

UNIX システムは、通常 USER 環境変数を定義し、その値をシステムに記録する ために利用する名前に設定します。従って、USER システムを使用する場合、カ レントユーザとしてユーザを確認するために Simulink では特に何も設定する必 要はありません。一方、Windows システムでは、システムにインストールされ ている Windows のバージョンとスタンドアロンで動作しているかネットワーク に接続されているかとに依存して、Simulink が必要とする "ユーザ名"の環境変 数をいくつか指定するか、何も指定しないかが決まります。MATLAB コマンド getenv を利用して、環境変数のどれかが定義されているかどうかを確認することができます。たとえば、つぎのように MATLAB コマンドラインで入力すると、

getenv('user')

USER 環境変数が Windows システムに存在するかどうかを確認することができます。設定されていない場合、ユーザ自身で設定する必要があります。 Windows95 と 98 では、システムの autoexec.bat ファイルで、つぎの行を実行することで値を設定します。

set user=yourname

ここで、yourname はモデルファイルで確定したいユーザ名です。ファイル autoexec.bat を保存し、変更を反映させるためコンピュータを再起動します。

注意 autoexec.bat ファイルは、典型的にシステムハードディスクの c:\ ディレクトリに存在します。

Windows/NT では、Windows/NT **システムのプロパティ**ダイアログの 環境パネル を使って、(まだ設定されていない場合)USER 環境変数を設定します。

stem Properties	?
Startup/Shutdown General	Hardware Profiles User Profiles Performance Environment
<u>S</u> ystem Variables:	
Variable	Value
OS Os2LibPath Path PATHEXT PROCESSOR_AR	Windows_NT C:\WiNNT\system32\os2\dl; c:\usr\bind:\rcs\binnt;d:\emacs-20.3.1\bin; COM;:EXE;:BAT;:PL x86
User Variables for paul	k:
Variable MATLAB MSDevDir path TEMP TMP	Value d:\r11 D:\DevStudio\SharedIDE c:\usr\bindt:\res\binntd:\\emacs-20.3.1\bin; C:\TEMP
Variable: USER	
vajue: Jyourname	S <u>e</u> t <u>D</u> elete
	OK Cancel Apply

システムのプロパティダイアログボックスを表示するために、スタート > 設定 > コントロールパネル を選択して Control Panel をオープンします。USER 変数を設 定するために、変数 フィールドに USER を、値 フィールドにログイン名を入力 します。設定 をクリックして新規の環境変数を保存します。それから、ダイア ログボックスをクローズするために OK をクリックします。
モデルプロパティダイアログ

Model Properties ダイアログボックスを利用して、バージョン管理パラメータを 編集し、関連するオプションを設定することができます。ダイアログボックスを 表示するには、Simulink の File メニューから Model Properties を選択します。

🛃 Block Diagram I	Properties: vdp_modelinfo
Model Properties	Options History
Model version:	1.9
Creator:	Rick
Created:	July 4
Model descriptio	n.
	OK Cancel Apply

Model Properties Pane

モデルプロパティパネルから、つぎのバージョン管理パラメータを編集することができます。

Creator. このモデルの作成者の名前。Simulink は、このプロパティをモデルが作成されたときの USER 環境変数の値に設定します。値を変更するために、このフィールドを編集できます。

Created. このモデルが作成された日付と時間。

Model description. モデルに関する説明。

Options Pane

Options パネルからコンフィグレーションマネージャを選択し、バージョン管理 情報の書式を指定することができます。

Block Diagram Propert Model Properties Opti	ies: vdp_modelinf	0			
Configuration manager: (For Model Info block)	Default (none)				
Model version format:	1.% <autoincremen< td=""><td>it:9></td><td></td><td></td><td></td></autoincremen<>	it:9>			
"Modified by" format: "Modified date" format:	% <auto></auto>				-11
L					
		OK	Cancel	Арр	ly

Configuration manager. モデルの管理に使用する外部設定マネジャーツールの確 認。リストから管理マネジャーを選択すると、注釈のための Model Info ブロック の管理マネジャーから情報を取り込めます。このオプションを設定しても、モデ ルの管理マネジャーを決定したり、アクティブにしたりはできません。デフォル トの Configuration manager の設定は、デフォルト(なし)です。Model Info ブ ロックの情報表示は、管理マネジャーシステムからは利用できません。詳細は、 9-159 ページの "Model Info" を参照 してください。

Model version format. Model Properties パネルと Model Info ブロックでモデルバー ジョン番号を表示するために利用される書式。このパラメータの値は、任意のテ キスト文字列です。テキスト文字列は、タグ %<AutoIncrement:#> を含むことが できます。ここで、# は整数です。Simulink は、モデルのバージョン番号を表示 するときにタグを整数で置き換えます。たとえば、

1.%<AutoIncrement:2>

は、

1.2

と表示されます。Simulink は、モデルを保存するときに、#を1ずつ増加しま す。たとえば、

1.%<1.%<AutoIncrement:2>

は、モデルを保存するときに、

1.%<1.%<AutoIncrement:3>

になり、Simulink はモデルのバージョン番号を 1.3 と記録します。

"Modified by" フォーマット. History パネルと、ヒストリ記録、Model Info ブロッ クで "Last modified by" の値を表示するために利用する書式。このフィールドの 値は、任意の文字列です。文字列は、タグ %<Auto> を含むことができます。 Simulink は、このタグを USER 環境変数のカレント値で置き換えます。

"Modified date"フォーマット. History パネルと、ヒストリ記録、Model Info ブロッ クで "Last modified date" を表示するために利用される書式。このフィールドの値 は、任意の文字列です。文字列は、タグ %<Auto> を含むことができます。 Simulink は、このタグをカレントの日付と時間で置き換えます。

History Pane

History パネルを使って、モデル更新履歴を表示、変数することができます。

Block Diagram Properti Model Properties Optic	es: vdp_modelinfo ns History	
Last modified by: Last modified date: Modified history update: Modified history: Edit	PaulK Mon Jul 27 10:15:58 1998 Prompt For Comments When Save PaulK Thu Jul 23 16:08:31 1998 Changed creator. PaulK Thu Jul 23 15:25:39 1998 Added ModelInfo block.	×
	OK Cancel	Apply

Last modified by. このモデルを最後に更新した人の名前。Simulink は、このパラ メータの値をモデルを保存するときの USER 環境変数の値に設定します。この フィールドを編集することはできません。

Last modified date. このモデルが最後に更新された日付。Simulink はこのパラメー タの値をモデルを保存するときのシステムの日付と時間に設定します。この フィールドを編集することはできません。

Modified history update. モデルを保存するときに、ユーザにコメントを促すかど うかを指定します。"Prompt for Comments When Save" を選択すると、Simulink は モデルに記録しているコメントをユーザに促します。現在のセッションでのモデ ルの変更点をドキュメント化するためにコメントを利用します。Simulink は、モ デルの更新履歴にこのパラメータの以前の値を記録します。詳細は、4–113 ペー ジの"モデル更新履歴の作成"を参照してください。

Modified history. このモデルの更新履歴。Simulink は、モデルを更新するときに ユーザが入力したコメントを履歴にまとめます。Edit をクリックすると、いつ でも履歴を編集することができます。

モデル更新履歴の作成

Simulink は、モデル更新に関する記録を作成し、モデル自身に記録することができます。Simulink は、複数ユーザがモデルを更新し保存するときに入力したコメントから自動的に履歴をまとめあげます。

更新の記録

更新履歴を始めるために、Simulinkの Model Properties ダイアログの History パネルから Modified history update オプションに対して "Prompt For Comments When Save" オプションを選択します。つぎに、モデルを保存すると、Simulink は Log Change ダイアログを表示します。

🛃 Log Change: vdp_modelinfo	×
Modified Comment:	
PaulK Mon Jul 27 17:22:51 1998	
 Show this dialog box next time when save Include "Modified Comments" in "Modified History" 	Save

モデルの変更履歴に項目を追加するためには、Modified Comments エディット フィールドに項目を入力し、Save をクリックします。このセッションに関する 項目を入力したくない場合は、Include "Modified Contents" in "Modified History" オプションの選択を解除します。更新の記録を中止したい場合は、 Show this dialog box next time when save オプションの選択を解除します。

更新履歴の編集

モデルの更新履歴を編集するために、Simulink の Model Properties ダイアログ ボックスの History パネルの Edit ボタンをクリックします。Simulink は、 Modification History ダイアログボックスにモデルの履歴を表示します。

🛃 Modification History:	vdp_modeli	info		_ 🗆 ×
Modified history:				
PaulK Mon Jul 27 17:22:	51 1998			
PaulK Thu Jul 23 16:08: Changed creator.	31 1998			
PaulK Thu Jul 23 15:25: Added Modelinfo block.	39 1998			
		OK	Cancel	Apply

ダイアログに表示される履歴を編集し、Apply または OK を選択して変更を保存 します。

バージョン管理プロパティ

Simulink は、モデルパラメータとしてバージョン管理情報をモデルに記録しま す。MATLAB コマンドラインや M ファイルから Simulink の get_param コマンド を使ってこの情報にアクセスできます。つぎの表は、バージョン管理情報を記録 するために Simulink が利用するモデルパラメータを一覧します。

プロパティ	内容
Created	作成した日付
Creator	モデルを作成した人の名前
ModifiedBy	モデルを最後に更新した人
ModifiedByFormat	ModifiedBy パラメータの書式。値は、文字列 です。文字列は、タグ % <auto> を含むこと ができます。Simulink はタグを USER 環境変 数のカレント値で置き換えます。</auto>

プロパティ	内容		
ModifiedDate	更新日		
ModifiedDateFormat	ModifiedDate パラメータの書式。値は文字列 です。文字列は、タグ % <auto> を含むこと ができます。Simulink は、モデルを保存した ときににタグを現在の日付と時間で置き換え ます。</auto>		
ModifiedComment	モデルを最後に更新したユーザが入力したコ メント		
ModifiedHistory	モデルの更新の履歴		
ModelVersion	バージョン番号		
ModelVersionFormat	モデルのバージョン番号の書式。この値は文 字列です。文字列は、タグ % <autoincrement:#>を含むことができます。 ここで、# は整数です。Simulink は、バー ジョン番号を表示するときにタグを # で置き 換えます。モデルを保存するときに # はイン クリメントされます。</autoincrement:#>		
Description	モデルに関する記述		
LastModificationDate	最終更新日		

Simulink セッションの終了

Simulink セッションは、すべての Simulink ウィンドウをクローズすることに よって終了します。

File メニューからつぎのコマンドを選択することによって MATLAB セッション を終了します。

- Microsoft Windows システムの場合: Exit MATLAB
- UNIX システムの場合: Quit MATLAB

シミュレーションの実行

はじめに						. 5-2 . 5-2 . 5-3
メニューコマンドを用いたシミュレーションの実行 シミュレーションパラメータの設定とソルバの選択 シミュレーションパラメータの適用 シミュレーションの開始						. 5-4 . 5-4 . 5-4 . 5-4
Simulation Diagnostics ダイアログボックス	•	•	•	•	•	. 5-6
Simulation Parameters ダイアログボックス						. 5-8 . 5-8 5-18 5-25 5-29
シミュレーションの性能と精度の改良						.5-34 5-33 5-35
コマンドラインからのシミュレーションの実行 sim コマンドの利用						.5-36 5-36 5-35

はじめに

シミュレーションは、Simulinkのメニューコマンドを使用するかまたは MATLAB コマンドウィンドウにコマンドを入力することによって実行すること ができます。

多くのユーザは、モデルを開発し、より洗練する際にはメニューコマンドを使用し、"バッチ"モードでシミュレーションを実行する際には MATLAB コマンドウィンドウにコマンドを入力します。

メニューコマンドの使用法

メニューコマンドを使ってシミュレーションを実行するのは簡単で、対話型に行われます。これらのコマンドにより、コマンド構文を覚える必要なく、ODE ソルバを選択し、シミュレーションパラメータを定義することができます。重要な利点は、シミュレーションを実行している間、特定の操作を対話型で実行することができるということです。

- 終了時間、ソルバ、最大ステップサイズなど多くのシミュレーションパラ メータを変更することができます。
- ソルバを変更することができます。
- 同時に別のシステムをシミュレーションすることができます。
- ラインをクリックすることにより、そのライン上を伝搬される信号を、フローティング(結線されていない)Scope ブロックまたは Display ブロック上で見ることができます。
- つぎの変更が行われない限り、ブロックのパラメータを修正することができます。
 - 状態、入力、出力の数
 - サンプル時間
 - ゼロクロッシングの数
 - 任意のブロックパラメータのベクトル長
 - 内部ブロックの作業ベクトルの長さ

シミュレーション実行中に、ラインやブロックを追加したり、削除するなど、モ デル構造への変更を加えることはできません。これらの変更を行う必要がある場 合には、シミュレーションを停止し、変更を行い、それからシミュレーションを 再開して変更の結果を見る必要があります。

コマンドラインからのシミュレーションの実行

コマンドラインからシミュレーションを実行することは、シミュレーションとブ ロックのパラメータを対話型で変更することができます。詳細については、5-36 ページの"コマンドラインからのシミュレーションの実行"を参照してください。

メニューコマンドを用いたシミュレーションの実行

本節では、Simulink メニューコマンドと Simulation Parameters ダイアログボックスを使用してシミュレーションを実行する方法について説明します。

シミュレーションパラメータの設定とソルバの選択

Simulation メニューから Parameters を選択して、シミュレーションパラメータ を設定し、ソルバを選択します。Simulink は、Simulation Parameters ダイアログ ボックスを表示します。このダイアログボックスは、シミュレーションパラメー タを管理するつぎの3つの"ページ"を使用します。

- Solver ページでは、開始時間と終了時間を設定し、ソルバを選択してソルバパ ラメータを指定し、いくつかの出力オプションを選択することができます。
- Workspace I/O ページは、MATLAB ワークスペースとの入出力を管理します。
- Diagnostics ページでは、シミュレーション中に表示される警告メッセージの レベルを選択することができます。

ページ上で設定するパラメータを含み、ダイアログボックスの各ページの詳細に ついては、5-8 ページの "Simulation Parameters ダイアログボックス"で説明しま す。

パラメータは、定数、ワークスペース変数名、MATLAB 関数、数学演算子から なる有効な MATLAB 表現として指定することができます。

シミュレーションパラメータの適用

シミュレーションパラメータを設定し、ソルバを選択すると、それらをモデルに 適用することができます。ダイアログボックス最下部の Apply ボタンを押して、 パラメータをモデルに適用します。パラメータを適用してダイアログボックスを 閉じるには、Close ボタンを押します。

シミュレーションの開始

ソルバとシミュレーションパラメータをモデルに適用したら、シミュレーション を実行する準備ができました。Simulation メニューから Start を選択して、シ ミュレーションを実行します。キーボードのショートカット Ctrl-T を使用する こともできます。Start を選択すると、メニュー項目は Stop に変わります。

コンピュータはビープ音を発してシミュレーションの終了を知らせます。

注意 Simulink を使いはじめたユーザが陥り易い誤りは、Simulink ブロックライ プラリがアクティブウィンドウであるときにシミュレーションを開始することで す。シミュレーションを開始する前にモデルウィンドウがアクティブウィンドウ であることを確認してください。

シミュレーションを停止するには、Simulation メニューから Stop を選択します。 シミュレーションを停止するためのキーボードのショートカットは、シミュレー ションを開始するのと同様に Ctrl-T です。

シミュレーションの実行は、Simulation メニューから Pause を選択することに よって中断することができます。Pause を選択すると、メニュー項目は Continue に変わります。中断したシミュレーションは、Continue を選択することによっ て継続することができます。

出力をファイルまたはワークスペースに書き出すようなブロックがモデルに含ま れている場合、あるいは Simulation Parameters ダイアログボックスで出力オプ ションを選択した場合には、Simulink はシミュレーションが終了または中断した ときにデータを書き出します。

Simulation Diagnostics ダイアログボックス

シミュレーション中にエラーが発生すると、シミュレーションが停止し、 Simulink は Simulation Diagnostics ダイアログボックスにエラーを表示します。



ダイアログボックスは、2つのパネルをもちます。上のパネルは、各エラーに対してつぎのような情報を表示する列で構成されます。

Message. メッセージのタイプ(例、ブロックエラー、ワーニング、ログ)

Source. エラーが生じたモデルの要素名(例、ブロック)

Fullpath.エラーが生じた要素のパス

Summary. 列に合わせて省略されたエラーメッセージ

Reported by. エラーを報告したコンポーネント (例、Simulink, Stateflow, Real-Time Workshop 等)

下のパネルは、初めは上のパネルに表示された最初のメッセージの全内容を含ん でいます。上のパネルの項目をシングルクリックすることにより、他のメッセー ジの内容を表示できます。

Simulation Diagnostics ダイアログボックスを表示するのに加えて、Simulink は(必要ならば)エラーのソースを含むブロック線図をオープンし、ソースを強調し ます。



同様に、上のパネルの対応するエラーメッセージをダブルクリックしたり、エ ラーメッセージ内のエラーのソース名(青で強調されている)をダブルクリック したり、またダイアログボックスの Open ボタンを選択することにより、エラー のソースを表示できます。

Simulation Parameters ダイアログボックス

本節では、Simulation Parameters ダイアログボックス上で、あるいは sim (5-38 ページの sim を参照) および simset(5-43 ページの simset を参照) コマンドを用い て指定するシミュレーションパラメータについて説明します。パラメータは、ダ イアログボックスのページに表示されるとおりに説明します。

つぎの表は、各ダイアログボックスのページの最下部に表示されているダイアロ グボックスボタンによって実行される動作をまとめたものです。

ボタン	動作
ОК	パラメータ値を適用し、ダイアログボックスを閉じます。シミュ レーションが実行される時、パラメータ値は直ちに適用されま す。
Cancel	パラメータ値を最後にダイアログボックスを開いたときの値に戻 し、ダイアログボックスを閉じます。
Help	ダイアログボックスページ用のヘルプテキストを表示します。
Apply	現在のパラメータ値を適用し、ダイアログボックスを開いたまま にします。シミュレーションが実行されるとき、パラメータ値は 直ちに適用されます。

表 5-1: Simulation Parameters ダイアログボックスのボタン

Solver ページ

Solver ページは、Simulation メニューから Parameters を最初に選択したとき、 あるいは Solver タブを選択したときに表示されます。

Solver ページでは、以下のことが行えます。

- ・ シミュレーションの開始時間と終了時間の設定
- ソルバの選択とパラメータの指定
- 出力オプションの選択

🛃 Simulation Parameters: vdp 📃 🗖 🗙
Solver Workspace I/O Diagnostics Advanced Real-Time Workshop
Simulation time Start time: 0.0 Stop time: 20
Solver options Type: Variable-step 🔽 ode45 (Dormand-Prince)
Max step size: auto Relative tolerance: 1e-3
Min step size: auto Absolute tolerance: 1e-6
Initial step size: auto
Output options
Refine output Refine factor: 1
OK Cancel Help Apply

シミュレーション時間

新しい値を Start time と Stop time のフィールドに入力することによって、シ ミュレーションの開始時間と終了時間を変更することができます。デフォルトの 開始時間は 0.0 秒で、デフォルトの終了時間は 10.0 秒です。

シミュレーション時間と実際のクロック時間は同じではありません。たとえば、 10秒間のシミュレーションの実行には通常10秒はかかりません。シミュレー ションの実行に必要な時間は、モデルの複雑さ、ソルバのステップサイズ、コン ピュータのクロック速度など多くの要素によって異なります。

ソルバ

Simulink モデルのシミュレーションには、連立常微分方程式 (ODE) の数値積分 が含まれます。Simulink は、そのような方程式のシミュレーションに対していく つかのソルバを提供します。ダイナミックシステムの挙動が多様であることか ら、ソルバの中には、特定の問題を解くために他のソルバより効率的なものがあ ります。正確かつ高速の結果を得るためには、ソルバを選択し、パラメータを設 定する際に注意が必要です。

可変ステップソルバと固定ステップソルバを選択することができます。*可変ス テップソルバは、シミュレーション*実行中にそのステップサイズを自動的に変 更することができます。可変ステップソルバには、エラー制御とゼロクロッシン グ検出が用意されています。*固定ステップソルバは、シミュレーション*実行中 に同じステップサイズを使用します。固定ステップソルバには、エラー制御とゼ ロクロッシング検出機能はありません。ソルバの詳細については Using MATLAB を参照してください。

デフォルトのソルバ ユーザがソルバを選択しないと、Simulink がモデルの状態 に応じてソルバを選択します。

- モデルが連続状態をもつ場合には、ode45 を使用します。ode45 は、すぐれた 汎用ソルバです。ただし、システムがスティッフであることが分かっており、 ode45 では適切な結果が得られない場合には、ode15s を試してみてください。 スティッフという言葉の定義については、後のページの注意、5-11 ページの "Variable-step solvers"を参照してください。
- モデルが連続状態をもたない場合には、Simulink は discrete と呼ばれる可変ス テップソルバを使用し、ode45 を使用していないことを示すメッセージを表示 します。Simulink は discrete と呼ばれる固定ステップソルバも提供します。つ ぎのモデルでは、2 つの discrete ソルバの違いが示されます。



0.5 および 0.75 のサンプル時間では、モデルの*基本サンプル時間* は 0.25 秒で す。可変ステップと固定ステップの discrete ソルバの違いは、それぞれが生成 する時間ベクトルにあります。

固定ステップ discrete ソルバは、つぎの時間ベクトルを生成します。 [0.0 0.25 0.5 0.75 1.0 1.25 ...]

可変ステップ discrete ソルバは、つぎの時間ベクトルを生成します。 [0.0 0.5 0.75 1.0 1.5 2.0 2.25 ...]

固定ステップ discrete ソルバのステップサイズは、基本サンプル時間です。可 変ステップ discrete ソルバは、可能な最大ステップを使います。

可変ステップソルバ 可変ステップソルバとして、ode45, ode23, ode113, ode15s, ode23s および discrete を選択することができます。デフォルトは、状態をもつシ ステムの場合は ode45、状態をもたないシステムの場合は discrete です。

- ode45 は、Dormand-Prince の陽的 Runge-Kutta(4,5) 公式に基づきます。これは単 段階 ソルパです。すなわち、y(t_n)の計算では直前の時間点 y(t_{n-1}) における解 のみを必要とします。一般に、ode45 は、ほとんどの問題に対して "最初の試み "として利用すべき最良のソルバです。
- ode23 もまた、Bogacki と Shampine の陽的 Runge-Kutta(2,3) に基づいています。大まかな許容値で中間のスティッフ性がある場合、ode45 よりも効率的な可能性があります。ode23 は単段階ソルバです。
- ode113 は、可変次数の Adams-Bashforth-Moulton PECE ソルバです。厳密な許容 値においては ode45 よりも効率的な可能性があります。ode113 は*多段階ソルバ* です。すなわち、現在の解を計算するためには、通常いくつかの過去におけ る解を必要とします。
- ode15s は、数値微分公式 (NDF) に基づく可変次数ソルバです。これらは、後退 微分公式 (BDF)(Gear 法としても知られている)と関連していますが、より効 率的です。ode113 と同様、, ode15s は多段階ソルバです。問題がスティッフな場 合、あるいは ode45 がうまくいかない、または効率が悪い場合には、ode15s を 試してみてください。
- ode23s は、2次の修正 Rosenbrock 公式に基づいています。これは単段階ソルバなので、大まかな許容値において ode15s より効率的な可能性があります。
 ode15s では効果的でないある種のスティッフな問題を解くことができます。
- ode23tは、"フリー"補間を利用して台形法を実行します。問題が適度にス ティッフで数値的減衰のない解を必要とする場合にこのソルバを利用してみ てください。
- ode23tb は、第一段階で台形法ステップを用い、第二段階で2次の後退差分公式 を利用する陰的 Runge-Kutta 法、TR-BDF 2を実行します。構造により、同じ イタレーション行列が両段階での実行に利用されます。ode23s のように、このソルバは大まかな許容値において ode15s より効率的な可能性があります。
- discrete(可変ステップ)は、モデルが連続状態をもたないことを検出したときに、 Simulinkが選択するソルバです。

注意 スティッフな問題の場合、解は積分間隔に比較して非常に短い時間スケー ルで変化する可能性がありますが、重要な解はそれよりはるかに長い時間スケー ルで変化します。スティッフな問題向きに考えられていない手法は、可能な限り 高速の変化を解くために十分なだけ小さな時間ステップを使用するので、解が ゆっくりと変化する区間に対しては効果的ではありません。ode15s と ode23s に 対しては、ヤコビアン行列が数値的に生成されます。詳細については、 Shampine, L. F. の *Numerical Solution of Ordinary Differential Equations*, Chapman & Hall, 1994 を参照してください。

固定ステップソルバ 固定ステップソルバとして、ode5, ode4, ode3, ode2, ode1, および discrete から選択することができます。.

- ode5 は、Dormand-Prince 公式である ode45 の固定ステップバージョンです。
- ode4 は、4 次 Runge-Kutta 公式である RK4 です。
- ode3 は、Bogacki-Shampine 公式である ode23 の固定ステップバージョンです。
- ode2 は、改良 Euler 公式としても知られる Heun の方法です。
- ode1 は、Euler の方法です。
- discrete(固定ステップ)は、積分を行わない固定ステップソルバです。状態をもたず、ゼロクロッシング検出とエラー制御が重要でないモデルに適しています。

シミュレーションで満足できない結果を生じている恐れがある場合には、5-34 ページの"シミュレーションの性能と精度の改良"を参照してください。

ソルバオプション

デフォルトのソルバパラメータは、ほとんどの問題に対して正確かつ効率的な結 果を提供します。しかし、場合によっては、パラメータを調整すると性能が改良 されることがあります(パラメータの調整については、5-34 ページの"シミュ レーションの性能と精度の改良"を参照してください)。選択したソルバは、 Solver パネルのパラメータ値を変更することによって調整することができます。

ステップサイズ

可変ステップソルバの場合、最大ステップサイズと推奨初期ステップサイズのパ ラメータを設定することができます。デフォルトでは、これらのパラメータは値 autoによって示されるように自動的に決定されます。

固定ステップソルバの場合、固定ステップサイズを設定することができます。デフォルトは auto です。

最大ステップサイズ. Max step size パラメータは、ソルバが使うことのできる最大の時間ステップを制御します。デフォルト値は、開始時間と終了時間から決定されます。

$$h_{max} = \frac{t_{stop} - t_{start}}{50}$$

ー般に、デフォルトの最大ステップサイズで十分です。ソルバの重要な挙動を捉 えていない恐れがある場合には、パラメータを変更してソルバがあまり大きなス テップを使わないようにすることができます。シミュレーション時間が非常に長 い場合、デフォルトのステップサイズは、ソルバが解を見つけるには大きすぎる 場合があります。また、モデルに周期的、または周期的に近い挙動が含まれ、そ の周期がわかっている場合には、最大ステップサイズをその周期の何分の1か (1/4 など)に設定してください。

一般に、出力点が多い場合は、最大ステップサイズではなくリファインファクタを変更します。詳細については、5-16ページの"リファインファクタ"を参照してください。

初期ステップサイズ.デフォルトで、ソルバは開始時間における状態の導関数 を検討することによって初期ステップサイズを選択します。最初のステップサイ ズが大きすぎると、ソルバは重要な挙動を飛び越す可能性があります。初期ス テップサイズパラメータは、*推奨する*最初のステップサイズです。ソルバはこ のステップサイズを試みますが、エラー基準が満たされなければそれを小さくし ます。

最小ステップサイズ.ソルバが使うことのできる最小の時間ステップを制御す ることができます。ソルバがエラー許容値程の小さなステップを使う必要がある 場合、現在の効果的な相対許容値を示すワーニングを出力します。このパラメー タはゼロ以上の大きな実数か、あるいは2要素のベクトルにすることができま す。ここで2要素とは、1番目の要素が最小ステップサイズ、2番目の要素がエ ラーが出力される前に出力される最小ステップサイズワーニングの最大数です。 2番目の要素をゼロにすると、指定した最小値よりも小さなステップをソルバー が使用しなければならなくなった最初からエラーになります。これは、 Diagnostics パネルの最小ステップサイズ違反診断をエラーに変更することと同じ ことです。また、これは、入力がスカラの場合のデフォルトです。このパラメー タのデフォルト値は、だいたい、マシン精度やワーニングの無制限数程度の最小 ステップサイズです。

誤差許容値

ソルバは、標準のローカル誤差制御法を用いて時間ステップごとに誤差をモニタ します。各時間ステップの間、ソルバはステップの終端で状態値を計算し、ロー カル誤差、すなわちこれらの状態値の予測誤差も判別します。それから、ローカ ル誤差を、相対的許容値(*rtol*)と絶対的許容値(*atol*)の関数である許容誤差と比 較します。誤差が*任意*の状態に対する許容誤差より大きい場合には、ソルバは ステップサイズを小さくして再び試みます。

- 相対的許容値は、各状態のサイズに関して誤差を測定します。相対的許容値は、状態の値の割合を表します。デフォルトの 1e-3 は、計算された状態が 0.1%以内の精度であることを意味します。
- 絶対的許容値は、しきい値誤差値です。この許容値は、測定された状態の値 がゼロに近づくに連れて許容誤差を表します。

i番目の状態に対する誤差 eiは、次式を満足する必要があります。

 $e_i \leq max(rtol \times |x_i|, atol_i)$

下の図は、状態のプロットと、許容誤差が相対的許容値と絶対的許容値で決められる領域を示しています。



auto (デフォルト)を指定すると、Simulink は各状態に対する絶対許容誤差の初 期値を 1e-6 に設定します。シミュレーションが進行すると、Simulink は各状態 に対する絶対許容誤差を、状態に対する相対許容誤差と状態とが最もかけ離れる と仮定される最大値に再設定されます。従って、状態が0から1 に変化し reltol が 1e-3 の場合、シミュレーションの終わりに abstol も 1e-3 に設定されます。ま た、状態が0から1000 に変化するとき、abstol は1 に設定されます。

計算された設定値が適切でない場合、適切な設定値を指定することができます。 絶対的許容値に対する適切な値を決定するには、シミュレーションを複数回実行 する必要があります。状態の大きさが幅広く変化する場合、状態に応じて異なる 絶対的許容値を指定する方が適切かもしれません。これは、Integrator ブロック のダイアログボックスで行うことができます。

ode15s に対する最大次数

ode15s ソルバは、1次から5次までのNDF公式に基づいています。次数が高い 公式ほど精度は高くなりますが、安定性は低くなります。モデルがスティッフ で、かつ安定性を必要とする場合は、最大次数を2に設定します(NDF公式が A-安定である最高次数)。ode15s ソルバを選択すると、ダイアログボックスにこ のパラメータが表示されます。

もう1つの方法として、固定ステップの低次(および A-安定の)ソルバである ode23s ソルバを使用することもできます。

Multitasking オプション

固定ステップソルバを選択する場合、Simulation Parameters ダイアログボックス の Solver ページは、Mode オプションリストを表示します。リストを使って、つ ぎのシミュレーションモードのうちの1つを選択することができます。

MultiTasking. このモードは、ブロック間で不正なサンプルレート遷移、つま り、異なるサンプルレートにおいて機能するブロック間の直接接続を検出した場 合にエラーを発生します。リアルタイムマルチタスクシステムでは、タスク間の 不正なサンプルレートの遷移により、他のタスクによって要求されたときに、タ スクの出力が利用不可能になります。そのような遷移をチェックすることによ り、マルチタスクモードは、実世界のマルチタスクシステムの有効なモデル作成 に役立ちます。モデルの部分がコンカレントタスクを表わします。

<u>rate transition</u> ブロックを使って、モデルから不正なレート遷移を除去します。 Simulink は、2 つのブロック、Unit Delay(9-254 ページの Unit Delay を参照)と Zero-Order Hold (9-262 ページの Zero-Order Hold を参照)を提供します。不正な遅 いブロックから速いブロックへの遷移を除去するには、遅いブロックの出力端子 と速いブロックの入力端子の間で遅いレートで実行する Unit Delay ブロックを挿 入してください。不正な速いブロックから遅いブロックへの遷移を除去するに は、速いブロックの出力端子と遅いブロックの入力端子の間で遅いレートで実行 する Zero-Order Hold ブロックを挿入してください。詳細は、*Real-Time Workshop Users Guide* の第7章、"複数のサンプルレートをもつモデル"を参照してください。 SingleTasking. このモードは、ブロック間のサンプルレートの遷移をチェックしません。このモードは、シングルタスクシステムのモデリング時に役立ちます。 そのようなシステムにおいて、タスクの同期化は問題ではありません。

Auto. このオプションによって、Simulink は、すべてのブロックが同じサンプル レートで操作される場合はシングルタスクモードを、モデルが異なるレートで操 作されるプロックを含む場合はマルチタスクモードを利用します。

出力オプション

ダイアログボックスの Output options 領域では、シミュレーションが生成する出 力量を制御することができます。つぎの3つのポップアップオプションから選択 することができます。

- · Refine output
- Produce additional output
- · Produce specified output only

リファインファクタ.シミュレーション出力が粗すぎる場合、Refine output の 選択により、追加の出力点を提供します。このパラメータは、時間ステップ間に 整数数の出力点を追加します。たとえば、リファインファクタが2の場合、各時 間ステップでの出力と共に中間点での出力を提供します。デフォルトのリファイ ンファクタは1です。

出力をスムーズにするには、ステップサイズを小さくする代わりにリファイン ファクタを変更する方がはるかに高速です。リファインファクタを変更すると、 ソルバはそれらの点で連続的な拡張式を評価することによって追加点を生成しま す。リファインファクタを変更してもソルバが使用するステップは変わりませ ん。

リファインファクタは可変ステップソルバに適用され、ode45 を使用するときに 最も有効です。ode45 ソルバは、大きなステップを取ることができます。シミュ レーション出力をグラフ化する場合、このソルバからの出力が十分滑らかでない ことがあります。この場合には、リファインファクタを大きくしてシミュレー ションを再度実行します。値を4にするとよりかなり滑らかな結果が得られるは ずです。

注意 このオプションによって、ソルバのゼロクロッシングの検出機能が抑制されます (3-14 ページの"ゼロクロッシングの検出"を参照してください)。

追加出力の生成. Produce additional output を選択すると、ソルバが出力を生成す る追加的な時間を直接指定することができます。このオプションを選択すると、 Simulink は、Solver ページに Ouput Times を表示します。このフィールドに追加 する時間または追加する時間のベクトルを評価する MATLAB 表現を入力しま す。追加出力は、追加する時間における連続拡張式を使って作成されます。リ ファインファクタと異なり、このオプションは、シミュレーションのステップサ イズを変更するので、時間ステップは、追加出力に対してユーザが指定した時間 と一致します。

指定出力のみの生成 Produce specified output only を選択すると、指定した出力 時間でのみ シミュレーション出力を提供します。このオプションは、シミュ レーションのステップサイズを変更するので、時間ステップは出力の生成のため にユーザが指定した時間と一致します。この選択は、異なるシミュレーションと 比較するときに、シミュレーションが同時に出力を生成するようにする場合に有 効です。

出力オプションの比較 サンプルシミュレーションは、つぎの時間で出力を生成します。

0, 2.5, 5, 8.5, 10

Refine output を選択し、リファインファクタ2を指定すると、つぎの時間で出力を生成します。

0, 1.25, 2.5, 3.75, 5, 6.75, 8.5, 9.25, 10

Produce additional output オプションを選択し、[0:10] を指定すると、つぎの時間で出力を生成します。

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

そして、可変ステップソルバが選択したステップ時間に応じて、追加された時間 において、生成する場合もあります。

Produce Specified Output Only オプションを選択し、[0:10] を指定すると、つぎの時間で出力を生成します。

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

一般的には、積分時間を基礎ステップサイズとして、出力を基礎ステップサイズ として、出力を指定することをお勧めします。たとえば、

[1:100]*0.01

は、つぎよりも高精度です。

[1:0.01:100]

Workspace I/O ページ

シミュレーション結果をワークスペース変数に直接出力したり、ワークスペース から入力と初期状態を得ることができます。Simulation Parameters ダイアログ ボックスで、Workspace I/O タブを選択します。つぎのようなページが表示され ます。

Simulation Parameters	: thermo
Solver Workspace I/O Diagno:	stics Advanced Real-Time Workshop
Load from workspace	Save to workspace
🗖 Input: 🛛 🗍	Time: tout
🔲 Initial state: 🗍	States: xout
	🗖 Output: yout
	Final state: 🗐 🖂
Save options	0
Decimation:	
Format: Arra	y 🔽
k o	K Cancel Help Apply

ベースワークスペースから入力の読み込み

Simulink は、シミュレーションの実行中、モデルの最上位の入力端子にモデルの ベースワークスペースから入力を適用することができます。このオプションを指 定するためには、Workspace I/O ページの Load from workspace エリアで Input ボックスをチェックします。それから、(つぎに示す)外部入力仕様を隣接する 入力ボックスに入力し、Applyを選択します。

外部(すなわち、ワークスペースからの)入力は、つぎの形式のいずれかになり ます。

Array. この形式を用いるには、Load from workspace エリアで Input ボックスを チェックし Workspace I/O ページの Format リストから Matrix オプションを選択 します。このオプションを選択すると、Simulink は Input チェックボックスの隣 の表現を評価し、その結果をモデルへの入力として用います。

形式は、データタイプ double の実数(非複素数)行列に表現しなければなりません。外部入力行列の最初の列は、昇順の時間ベクトルでなければなりません。残りの列は、入力値を指定します。特に、各列が異なる(連続する順番の)Inport ブロック信号に対する入力を表現し、各行が関連する時間点での入力値になりま す。Simulink は、Interpolate data オプションが関連する入力に対して選択されているとき、必要であれば入力値を線形に内挿または外挿します。(9-123 ページの "Interpolate data"を参照してください)。

入力行列の列の総数は n + 1 である必要があり、ここで n はモデルの入力端子に 入力される信号の総数となります。

デフォルトの入力表現は [t,u] であり、デフォルトの入力形式は行列です。ベー スワークスペースで t と u が定義された後、モデルのベースワークスペースから データを取り込むためには Input オプションをチェックするだけです。たとえ ば、モデルが 2 つの入力端子を持ち、その内の 1 つが 2 つの信号を受けいれ、も う一方が 1 つの信号を受け入れると仮定してください。また、ベースワークス ペースでつぎのように u と t が定義されていると仮定してください。

t = (0:0.1:1)';

u = [sin(t), cos(t), 4*cos(t)];

注意 入力行列の要素は、タイプ double の実数(非複素数)値でなければなりま せん。複素数データの入力や、double 以外のデータタイプや行列 (2-D)の入力に は、構造体を使わなければなりません。 Structure with time. Simulink は、Input テキストフィールドで指定された名前の 構造体の形式でワークスペースからデータを読み込むことができます。入力構造 体は2つのトップレベルフィールド time と signals をもたなければいけません。 time フィールドはシミュレーション時間の列ベクトルです。signals フィールドは 構造体の配列で、おのおのがモデルの入力端子に関連します。各サプ構造は フィールド values をもっています。values フィールドは関連する入力端子に対す る入力の列ベクトルです。

各 signals サブ構造は、values と dimensions フィールドの2つのフィールドを もっていなければなりません。values フィールドは、対応する入力端子に対する 入力配列を含む必要があり、各入力は、time フィールドで指定される時間に相 当します。dimensions フィールドは、入力の次元を指定します。各入力がスカ ラーあるいはベクトル(1-D 配列)であれば dimensions フィールドは、ベクトル の長さ(スカラーの場合は1)を指定するスカラー値になります。各入力が行列 (2-D 配列)であれば、dimensions フィールドは、2 要素のベクトルになります。 1 番目の要素は、行列の行数を指定し、2 番目の要素は列数を指定します。

端子への入力がスカラーあるいはベクトル値であれば、values フィールドは M × N の配列となります。ここで、M は time フィールドで指定された時間点の数 であり、N は各ベクトル値の長さです。たとえば、つぎのコードは、単一入力端 子をもつモデルに、int8 型タイプの 2 要素信号ベクトルの 11 時間点分のサンプ ルをロードするための構造体を作成します。

a.time = (0:0.1:1)'; c1 = int8([0:1:10]'); c2 = int8([0:10:100]'); a.signals(1).values = [c1 c2]; a.signals(1).dimensions = 2;

モデルにこのデータをロードするには、Workspace I/O パネルの Input オプションをチェックし、入力形式のフィールドに a を入力します。

端子への入力が行列 (2-D 配列) の場合、値のフィールドは、MxN × T 配列にし なければなりません。ここで、M とN は各行列入力の次元で、T は時間点の数 です。たとえば、4 × 5 行列信号で 51 時間点サンプルをモデルの入力端子の 1 つに入力したい場合を仮定します。このとき、ワークスペース構造体の対応する 次元フィールドは、[4 5] に等しくし、値の配列は、4 × 5 × 51 次元にしなけれ ばなりません。 たとえば、2つの入力をもつ、つぎのモデルを考えます。



1番目の端子に正弦波、2番目の端子に余弦波を入力すると仮定します。これを 実現するために、ベースワークスペースにつぎのようなベクトル a を定義しま す。

a.time = (0:0.1:1)'; a.signals(1).values = sin(a.time); a.signals(1).dimensions = 1; a.signals(2).values = cos(a.time); a.signals(2).dimensions = 1;

そして、Input ボックスをチェックし、隣接したテキストフィールドに a と入力し、I/O フォーマットに Structure with time を選択します。

注意 Simulink は、**Structure with time** 出力フォーマットでワークスペースに保存 されたシミュレーションデータを再読み込みします。詳細は、5-23 ページの "Structure with time" を参照してください。

Structure. 構造体フォーマットは、time フィールドが空であることを除いて、 Structure with time フォーマットと同じです。たとえば、先ほどの例題でつぎの ように time フィールドを設定することができます。

a.time = []

この場合、Simulink は入力端子の値の配列の第1要素から最初の時間ステップに 対する入力を読み込み、値の配列の第2要素から2番目の時間ステップに対する 入力を読み込みます。その他も同様に読み込みます。

注意 Simulink は、Structure 出力フォーマットでワークスペースに保存されたシ ミュレーションデータを再読み込みします。詳細は、5-24 ページの "Structure"を 参照してください。 Per-Port Structures. このフォーマットは、時間付き構造体や各端子に対する時間なし構造体の分割から構成されます。各端子の入力データ構造は signals フィールドのみを持っています。このオプションを指定するために、Input テキ ストフィールドで、構造体の名前をコンマで分割したリスト in1, in2, ..., inN のよ うに入力します。ここで、in1 はモデルの最初の入力端子に対するデータ、in2 は 2 番目の入力端子に対するデータ、その他も同様に各入力端子に対するデータで す。

Time Expression.時間表現は、行ベクトルの長さがモデルの入力端子に入力される信号の数と等しくなる MATLAB 表現です。例えば、モデルが2つの信号を受け入れる1つのベクトル入力を持っていると仮定します。更に、timefcnが2 要素の行ベクトルを出力するユーザ定義関数と仮定します。つぎの記述は、このようなモデルに対して妥当な入力時間の表現です。

'[3*sin(t), cos(2*t)]'

'4*timefcn(w*t)+7'

Simulink はシミュレーションの各ステップにおいて表現を評価し、その結果の値 をモデルの入力端子に適用します。シミュレーションを実行しているとき、 Simulink が変数 t を定義するということに注意してください。また、1 つの変数 をもつ関数に対する表現で時間変数を省略することができます。たとえば、 Simulink は表現 sin を sin(t) として解釈します。

Saving Output to the Workspace

このダイアログボックスページの Save to workspace 領域内の Time, States, Output チェックボックスを選択することによって出力変数を指定することができます。 出力変数を指定すると、Simulink は(選択したのと同数の)時間、状態、出力軌 跡に対する値をワークスペースに書き出します。

異なる変数に値を割り当てるためには、チェックボックスの右側のフィールドに それらの変数名を指定します。複数の変数に出力を書き出すためには、コンマで 区切ったリストとして変数名を指定します。Simulink は、Save to Workspace 領域 で指定した名前のベクトルでシミュレーション時間を保存します。

注意 Simulink は、モデルのベースサンプルレートにおいて出力をワークスペー スに保存します。他のサンプルレートで出力を保存したい場合は、To Workspace プロックを使います。(9-240 ページの "To Workspace" を参照)。 Save options 領域では、フォーマットを指定し、保存する出力のデータ数を制限 することができます。

モデルの状態と出力に対するフォーマットオプションは、つぎの通りです。

Array. このオプションを選択すると、Simulink はモデルの状態と、状態の出力と出力配列をそれぞれ保存します。

状態行列は、Save to Workspace 領域で指定された名前(たとえば xout)をもつ行 列です。状態行列の各列はモデル状態の時間サンプルに対応し、各行は特定の時 間での状態に対応します。例えば、モデルが2つの連続する状態をもつとき、こ れらの状態は2要素のベクトルとなります。このとき、状態行列の各行のはじめ の2要素は、1番目の状態ベクトルの時間サンプルを含みます。各行の最後の2 要素は、2番目の状態ベクトルの時間サンプルを含みます。

モデル出力行列は、Save to Workspace 領域で指定された名前 (たとえば yout)を もちます。各列はモデル出力に関連し、各行は特定の時間での出力に関連しま す。

注意 モデルの出力と状態は、つぎの場合に、配列形式を使用して保存すること ができます。出力が、すべてスカラあるいはすべてベクトル(あるいはすべて状 態行列)の場合、また出力が、すべて実数あるいはすべて複素数である場合、ま たは、すべて同じデータタイプである場合。

Structure with time. この形式を選択すると、モデルの状態と出力が、Save to Workspace 領域で指定した名前をもつ構造体に保存されます。

構造体は、2 つのトップレベルフィールド time と signals をもちます。time フィー ルドは、シミュレーション時間のベクトルです。signals フィールドはサブ構造の 配列で、おのおの出力端子に対応しています。各サブ構造は、4 つのフィールド values, dimensions, label, blockName をもちます。values フィールドは、関連する出 力端子への出力値をもっています。出力がスカラーあるいはベクトルのとき、 values フィールドは行列であり、その行は time ベクトルの相当する要素で指定 される時間での出力をあらわします。出力が行列(2-D)である場合、value フィールドは M × N × T 次元の 3-D 配列です。M × N は出力信号の次元で あり、T は出力するサンプル数です。T=1 であれば、MATLAB は最後の次元を 無視します。label フィールドは、出力端子に接続された信号、あるいは(連続あ るいは離散)状態のタイプを指定します。 dimensions フィールドは、出力信号の次元を指定します。label フィールドは、出 力端子に接続される信号のラベルを指定します。blockName フィールドは、対応 する outport や状態を伴ったブロックの名前を指定します。

状態の保存に使用される構造体は、特定の構造が使われます。状態の構造体は、 2 つのトップラベルフィールド、time と signals をもっています。time フィール ドは、シミュレーション時間のベクトルを含みます。signals フィールドは、サ ブ構造体の配列を含んでおり、それぞれは、モデルの状態に対応します。各々の signals の構造体は、4 つのフィールド、values、demension、label、blockName を もちます。values フィールドは、blockName フィールドで指定されるブロックの 状態の時間サンプルを含んでいます。組込みブロックのための label フィールド は state のタイプ、CSTATE(連続状態)あるいは DSTATE(離散状態)を指定し ます。S-ファンクションブロックのためには、label は、S-ファンクションブ ロックによって状態に付けられたどんな名前も含みます。

状態の時間サンプルは、値の行列として values フィールドに保存されます。 各々の行は、時間サンプルに対応します。行の各要素は、状態の要素に対応して います。状態が行列の場合、行列は列の大きい順で values 配列に保存されます。 たとえば、モデルが2×2行列状態とシミュレーション実行の間に状態の51サ ンプルの Simulink ログを含んでいると仮定します。そのとき、この状態の values フィールドは、51×4行列をもちます。ここで、各行は状態のサンプル時間に 対応し、各行の最初の2要素はサンプルの1番目の列に対応します。また、サイ トの2要素は、サンプルの2番目の列に対応します。

Structure. このフォーマットは、Simulink が保存した構造体の time フィールド にシミュレーション時間をストアしないことを除いて、上記と同じになります。

Per-Port Structures. このフォーマットは、時間付き構造体や各出力端子に対す る時間なし構造体の分割から構成されます。各出力データ構造は signals フィー ルドのみをもちます。このオプションを指定するために、Output テキストフィー ルドで構造体の名前をコンマで区切られたリスト out1, out2, ..., outN のように入 力します。ここで、out1 はモデルの最初の端子に対するデータ、out2 は 2 番目の 端子に対するデータ、その他もそれぞれの端子に対するデータです。

保存するデータの行数を制限するには、Limit data points to last というラベルの付いたチェックボックスを選択し、保存する行数を指定します。間引きファクタを 適用するには、Decimation ラベルの右側のフィールドに値を入力します。たとえ ば、値2は生成される点を1つおきに保存します。

状態のロードと保存

シミュレーションの開始時にシステムに適用される初期条件は、一般にブロック 内で設定されます。ブロック内で設定された初期条件は、それらをこのページの States 領域に指定することによって変更することができます。

シミュレーションに対する最終状態を保存し、それらを別のシミュレーションに 適用することもできます。この機能は、定常状態の解を保存し、その既知状態で シミュレーションを再開したいときに有効です。状態は、WorkspaceI/O ページ の Save オプション領域で選択したフォーマットで保存されます。

最終状態 (シミュレーションが終了したときの状態値) は、Final State チェック ボックスを選択し、隣接するエディットフィールドに変数を入力することによっ て保存されます。

状態は、Initial State チェックボックスを選択し、初期状態値を含む変数の名前 を指定します。この変数は、行列か save final states で利用したのと同じ形式の構 造体です。これにより、Simulink は Structure や Structure with time 形式を利用 して、現在のセッションに対する初期状態を以前のセッションで保存された最終 状態に設定します。

チェックボックスが選択されていないか、状態ベクトルが空([])の場合、 Simulink はブロックに定義されている初期条件を使用します。

Diagnostics ページ

シミュレーション中に起こり得る種々のタイプのイベントや条件に対しては、 Simulation Parameters ダイアログボックス上の Diagnostics タブを選択すること によって、希望する動作を指示することができます。このダイアログボックス は、つぎのように表示されます。

Simulation Parameters: vdp	D - X
Solver Workspace I/O Diagnostics	Advanced Real-Time Workshop
Simulation options	
Consistency checking: off	Bounds checking: off
Configuration options:	Action
-1 sample time in source	Warning Action
Algebraic loop	Warning
Block priority violation	Warning C Warning
Check for singular matrix	None C Error
Data overflow	Warning
int32 to float conversion	Warning
Min step size violation	Warning Emman
S-function ungredes needed	None
Signal label migmatch	None I
, SIGNAL TADEL MISMACON	None III
ОК	Cancel Help Apply

ダイアログボックスはつぎのオプションをもちます。

整合性のチェック

適合性のチェックは、Simulinkの ODE ソルバによる特定の仮定を検証するデ バッグツールです。それを使用する主な目的は、S-ファンクションが Simulink の組み込みブロックと同じ規則に従うようにすることです。適合性のチェックを 行うと性能が著しく低下するため(最大 40%)、通常は off に設定しておく必要が あります。適合性のチェックは、S-ファンクションを検証し、予期せぬシミュ レーション結果の原因の判別に役立てるために使用します。

効率的な積分を実行するために、Simulink はある時間ステップの特定の値をつぎ の時間ステップで使用するために保存(キャッシュ)します。たとえば、時間ス テップの終端での微係数は、一般につぎの時間ステップの先頭で再利用すること ができます。ソルバはこれを利用して、無駄な微係数計算を行いません。

適合性のチェックのもう1つの目的は、与えられたt(時間)の値で呼び出され たときに、ブロックが一定の出力を生成できるように保証することです。ヤコビ アンの計算中、ブロックの出力関数を同じtの値で何回も呼び出すことができる ので、これはスティッフなソルバ(ode23s および ode15s)に対して重要です。

適合性のチェックが有効な場合、Simulink は適切な値を再計算し、それらを キャッシュされている値と比較します。値が同じでないと、エラーが発生しま す。Simulink は、つぎの計算値を比較します。

- 出力
- ゼロクロッシング
- 微係数
- 状態

範囲のチェック

このオプションによって、Simulink は、ブロックがシミュレーション中に割り当 てられたメモリ外に書き込んでいないかどうかをチェックします。一般的には、 バグがあるユーザ定義 S-ファンクションを含んだモデルの場合にのみ、この現 象が起こります。このオプションが利用可能であれば、ブロックが実行される毎 にこのチェックが実行されます。結果として、このオプションを利用可能にする と、モデルの実行速度がかなり遅くなります。したがって、モデルの実行を不必 要に遅くしないようにするために、モデルがバグのあるユーザ定義 S-ファンク ションを含んでいると思われる場合のみ、このオプションを利用してください。

設定オプション

この設定は、モデルの実行中に起こるイベントの異常タイプをリストします。 各々のイベントタイプでは、メッセージなし、ワーニングメッセージを出力、あ るいは、エラーメッセージを出力するかどうかを選択することができます。ワー ニングメッセージではシミュレーションは終了しませんが、エラーメッセージで は終了します。

イベント	説明
ソース内でサンプル時 間が -1	ソースブロック (Sine Wave ブロック) に -1 のサンプ ル時間が指定されます。
代数ループ	Simulink はモデルのシミュレーション中に代数ルー プを検出します。詳細は 3–18 ページの " 代数ループ " を参照してください。
特異行列のチェック	Product ブロックの逆行列の計算中に特異行列を検知 します (9-170 ページの Product 参照)。
データオーバーフロー	信号やパラメータの値が大きすぎて信号やパラメー タのデータタイプで表現できません。詳細は4-45 ページの"データタイプの機能"を参照してください。

イベント	説明 (Continued)
int32を float に変換	32-bit 整数値が浮動小数値に変換されました。変換に よって精度が落ちます。詳細は 4-45 ページの"データ タイプの機能"を参照してください。
最小ステップサイズの 違反	つぎのシミュレーションステップが指定した最小ス テップサイズよりも小さいです。これは、モデルの エラー許容値が指定した最小ステップサイズよりも 小さいステップサイズが要求されたときに起こりま す。詳細は、5-14 ページの"誤差許容値"や 5-14 ページの"誤差許容値"を参照してください。
マルチタスクレート変 換	マルチタスクモード中に 2 つのブロック間で起こっ た不正レート変換 (5-15 ページの "Multitasking オプ ション " 参照)
S-function アップデート の必要性	ブロックが現バージョンの機能を使用するために アップデートされていない S-function を検知。
信号ラベルのミスマッ チ	共通ソース信号をもつが異なるラベルをもったバー チャル信号をシミュレーション中に検知。(4-31 ペー ジの " バーチャル信号 ")
シングルタスクレート 変換	シングルタスクモードで動作中に、2 つのブロック間 でレート変換が行われた (5-15 ページの "Multitasking オプション ")。
接続されていないブ ロック入力	モデルは、接続されていない入力端子をもったブ ロックを含んでいます。
接続されていないブ ロック出力	モデルは、接続されていない出力端子をもったブ ロックを含んでいます。
接続されていないライ ン	モデルは、接続されていないラインを含んでいます。
不必要なタイプ変換	データタイプ変換ブロックが、タイプ変換を必要と しないところで使用されています。詳細は、9-49 ページの Data Type Conversion を参照してください。
イベント	説明 (Continued)
-------------------	--
ベクトル / 行列変換	ベクトルから行列、あるいは行列からベクトルへの 変換がブロック入力で起こっています (4–36 ページ の " ベクトルあるいは行列入力変換規則 " 参照)。
ブロックプライオリ ティ違反	モデルのシミュレーション中にブロックプライオリ ティ違反を検知しました。

アドバンスドパネル

アドバンスドパネルでは、シミュレーションの性能に影響を与えるオプションを 設定することができます。

🛃 Simulation Parameters: vdp	_ 🗆 X
Solver Workspace I/O Diagnostics Advanced Model parameter configuration Inline parameters Configure	
Optimizations: Block reduction Off Boolean logic signals Off Parameter pooling On Signal storage reuse On Zero crossing detection On	Action C On C Off
OK Cancel Help	Apply

モデルパラメータの設定

インラインパラメータ.デフォルトでは、多くのブロックのパラメータがシ ミュレーション中に調整("チューニング")可能です(3-5ページの"調整可能 なパラメータ"を参照)。このオプションを選択すると、ユーザが指定したパラ メータ以外のパラメータを調整不可能にします。パラメータを調整不可能にする と、Simulink は、パラメータを不変値として取り扱うため、シミュレーションの 速度が向上します。このオプションが選択されても、調整可能なパラメータとし て残しておきたいパラメータを指定するには、モデルパラメータ設定ダイアログ ボックス(5-32ページの"モデルパラメータ設定ダイアログボックス"を参照)を 使用します。パラメータを指定するダイアログボックスを表示するには、モデル パラメータ設定にある**設定**ボタンを選択してください。

このオプションを選択すると、シミュレーション中に調整可能なパラメータは、 つぎの条件を満たすパラメータのみとなります。

- パラメータの値は、MATLABのワークスペース上に定義されている変数でなければなりません。
- パラメータは、モデルパラメータ設定ダイアログボックスでグローバル(チューナブル)なパラメータとして指定されていなければなりません。

上記の条件を満たすパラメータを調整するには、対応するワークスペース変数の 値を変更し、Simulinkの編集メニューからモデルの更新(Ctrl+D)を選択してくだ さい。

このオプションを選択すると、Simulink は一定信号をシミュレーションループの外に出します。そのため、シミュレーションの速度が向上します。

最適化

ブロックリダクション.モデルを高速化するために、あるブロックグループを 統合したブロックに置換えます。

Boolean 論理信号. ブロックが Boolean 論理信号を要求します。このオプション をオフにすると、boolean タイプの信号を受け入れるブロックは、double タイプ の入力も受け入れます。例として、以下のモデルを考えてみます。



このモデルは、double タイプ信号を boolean タイプの入力も受け入れる論理演算 ブロックに統合します。Boolean 論理信号オプションがオンの場合、このモデル は実行時にエラーを出力します。Boolean 論理信号オプションがオフの場合、こ のモデルはエラーを出力せずに実行されます。 注意 このオプションにより、double タイプの信号のみをサポートしていた以前 の Simulink バージョンで作られたモデルを、現バージョンの Simulink で実行す ることが可能です。

パラメータプール.このオプションは、コード生成時に使用されます(より詳細 な情報は、Real-Time Workshopのマニュアルを参照してください)。コード生成 を行わないのであれば、このオプションをオンのままにしてください。

信号ストレージの利用.ブロックの入出力信号を蓄えるために割り当てられる メモリバッファを Simulink が再利用するかどうかを決定します。このオプショ ンがオフの場合、Simulink は各々のブロック出力のために、別々のメモリバッ ファを割り当てます。これによって、大規模モデルのシミュレーションに必要な メモリ量が実質的に増えてしまいます。そのため、このオプションは、モデルを デバッグする必要があるときだけ選択することをお勧めします。特に、つぎのよ うな時には信号ストレージの再利用をオフにすることを推奨します。

- C-MEX S-function のデバッグ。
- デバッグ中のモデルで信号を調べるためにフローティング Scope や Display ブロックを使用するとき。

Simulink は、信号ストレージの再利用を利用可能にした状態で、信号のバッファが再利用されている信号をフローティング Scope や Display ブロックで表示しようとすると、エラーダイアログボックスを開きます。

ゼロクロッシングの検知.可変ステップでシミュレーション中のモデルのゼロ クロッシングの検知を可能にします。大抵のモデルは、ソルバの時間ステップを より大きくすることで、シミュレーションの速度が向上します。モデルにかなり ダイナミックな変化があるような場合、このオプションを無効にすることでシ ミュレーションの速度を向上させることが可能ですが、シミュレーション結果の 精度が悪くなります。詳細な情報は、3-14 ページの"ゼロクロッシングの検出" を参照してください。

モデルパラメータ設定ダイアログボックス

モデルパラメータ設定ダイアログボックスより、選択されたパラメータのインラ インパラメータオプション(5-29ページの"モデルパラメータの設定"を参照)を 上書きすることができます。

Model Parameter Configuration:	vdj	0		_ 🗖
Description				
Define the global (tunable) parameters for your m 1, the simulation by providing the ability to tune 2, the generated code by enabling access to para	node para ame	el. These paramete ameters during exec ters by other modul	rs affect: sution, and es.	
Source list		Global (tunable)	parameters	
MATLAB workspace		Name	Storage class	Storage type qualifier
Name 1 balance				
2 gain				
Refresh list		1		New Remove
			OK Cancel	Help Apply

ダイアログボックスではつぎの項目の制御が行えます。

ソースリスト.ワークスペースの変数をリスト表示します。つぎのオプションがあります。

- MATLAB ワークスーペース
 MATLAB ワークスペースに存在するすべての数値変数をリストします。
- Referenced workspace variables

モデルが参照している変数のみをリストします。

リフレッシュリスト.ソースリストを更新します。リストを表示した後にワークスペースに変数を追加したときに、このボタンをクリックしてください。

表示追加.ソースリストで選択した変数をチューナブルパラメータテーブルに 追加します。

新規.新しいパラメータを定義し、チューナブルパラメータのリストに追加し ます。このボタンは、まだ MATLAB ワークスペースに定義されてない調整可能 なパラメータを作成するときに使用します。 注意 このオプションは、MATLAB ワークスペース内に対応する変数を作成しま せん。変数は、ユーザ自身で作成する必要があります。

ストレージクラス.コード生成に使用されます。詳細は、Real-Time Workshop のマニュアルをご覧ください。

ストレージタイプクオリファ.コード生成に使用されます。詳細は、Real-Time Workshop のマニュアルをご覧ください。

シミュレーションの性能と精度の改良

シミュレーションの性能と精度は、モデル設計、シミュレーションパラメータの 選択など、多くの条件によって影響を受けます。

ソルバは、ほとんどのモデルシミュレーションを、それらのデフォルトパラメー タ値で正確かつ効率的に処理します。しかし、モデルによっては、ソルバとシ ミュレーションのパラメータを調整すると、さらによい結果が得られる場合があ ります。また、モデルの挙動についての情報を知っている場合、この情報をソル バに提供すると、シミュレーションの結果を改良することができます。

シミュレーションの高速化

シミュレーションが遅いことには多くの原因があります。以下にそのいくつかを 示します。

- モデルに MATLAB Fcn ブロックが含まれています。モデルに MATLAB Fcn ブロックが含まれている場合、各時間ステップごとに MATLAB インタプリタが呼び出され、シミュレーションの速度が大幅に遅くなります。できるだけ組み込み Fcn ブロックまたは Elementary Math ブロックを用いてください。
- モデルに M-ファイルによる S-ファンクションが含まれています。M-ファイル による S-ファンクションによっても、MATLAB インタプリタが各時間ステッ プごとに呼び出されます。S-ファンクションをサブシステムか C-MEX ファイ ル S-ファンクションに変換することを考慮してください。
- モデルに Memory ブロックが含まれています。Memory ブロックを使用すると、 可変次数ソルバ (ode15s および ode113) は各時間ステップごとに1次に再設定 されます。
- ・最大ステップサイズが小さすぎます。最大ステップサイズを変更している場合には、デフォルト値(auto)で再度シミュレーションを実行してみてください。
- 非常に高い精度を要求していませんか?デフォルトの相対許容値(0.1% 精度)
 で、普通は十分な精度が得られます。また、状態量がゼロに近付くようなモデルでは、絶対許容値パラメータが小さすぎると、ゼロ付近の状態量でステップ数がかなり増えます。エラーの説明に関しては、5-14ページの"誤差許容値"を参照してください。
- 時間スケールが長すぎる可能性があります。時間区間を短くしてください。
- 問題がスティッフである可能性がありますが、ノンスティッフなソルバを使用しています。ode15sを使用してみてください。

- モデルは、互いに倍数でないサンプル時間を使用しています。互いに倍数でないサンプル時間が混在すると、ソルバはすべてのサンプル時間に対してサンプル時間を確実にヒットするために、結果として十分小さなステップを使うことになります。
- モデルに代数ループが含まれています。代数ループに対する解は、各時間ステップごとに繰り返し計算されます。したがって、性能が大幅に低下します。
 詳細については、3-18ページの"代数ループ"を参照してください。
- モデルは、Random Number ブロックを Integrator に入力しています。連続システムの場合、Sources ライブラリの Band-Limited White Noise ブロックを使用してください。

シミュレーション精度の改良

シミュレーションの精度をチェックするためには、妥当な時間区間でシミュレー ションを実行します。つぎに、相対的許容値を 1e-4 に下げるか (デフォルトは 1e-3)、絶対的許容値を小さくして再び実行します。両方のシミュレーションの 結果を比較します。結果にあまり違いがなければ、解が収束したことを確信する ことができます。

シミュレーションがその開始時での重要な挙動を見逃すような場合には、初期ス テップサイズを小さくし、シミュレーションが重要な挙動を "飛び越さない"よ うにします。

シミュレーション結果が時間の経過と共に不安定になる場合は、

- システムが不安定な可能性があります。
- ode15sを使用している場合には、最大次数を2に制限するか(ソルバがA-安定であるための最大次数)、ode23s ソルバを使用してみる必要があります。

シミュレーション結果が正確でないように見える場合は、

- 値がゼロに近づく状態をもつモデルの場合、絶対的許容値パラメータが大き すぎると、シミュレーションがゼロに近い状態値の周辺領域で使うステップ 数が少なすぎます。このパラメータ値を小さくするか、Integrator ダイアログ ボックスで個々の状態に対してそれを調整してください。
- 絶対的許容値を小さくしても精度が十分改良されない場合には、相対的許容値パラメータのサイズを小さくして許容誤差を小さくし、ステップサイズをあえて小さくすると共にステップ数を増やします。

コマンドラインからのシミュレーションの実行

シミュレーションコマンドを MATLAB コマンドウィンドウに入力したり、M-ファイルから入力して、シミュレーションを実行することができます。パラメー タをランダムに変更し、シミュレーションをループ内で実行することによって、 モンテカルロ解析を実行することができます。sim コマンド、または set_param コマンドを使ってコマンドラインからのシミュレーションを実行することができ ます。その両方のコマンドについて以下に説明します。

sim コマンドの利用

シミュレーションを実行するコマンドの完全な構文は、つぎのとおりです。

[t,x,y] = sim(model, timespan, options, ut);

model パラメータのみが必須です。コマンド上で与えられないパラメータには、 Simulation Parameters ダイアログボックスの設定値が使われます。

sim コマンドの詳細な構文については、5-38 ページの sim を参照してください。 options パラメータはソルバ名と誤差許容値を含む、追加的なシミュレーション パラメータを提供する構造体です。simset コマンド(5-43 ページの simset を参照) を用いて、options 構造体にパラメータを定義します。シミュレーションパラ メータについては、本章のはじめの部分 5-8 ページの "Simulation Parameters ダイ アログボックス"で説明してあります。

set_param コマンドの利用

シミュレーションの開始や終了、一時終了と続行を実行したり、ブロック線図を 更新するために set_param コマンドが利用できます。同様に、シミュレーション の状況を確認するために get_param コマンドが利用できます。このような目的の ための set_param コマンドのフォーマットはつぎのようになります。

set_param('sys', 'SimulationCommand', 'cmd')

ここで、'sys' はシステムの名前で、'cmd' は 'start', 'stop', 'pause', 'continue', 'update' です。

この目的のための get_param コマンドのフォーマットは、つぎのようになります。

get_param('sys', 'SimulationStatus')

Simulink は、'stopped', 'initializing', 'running', 'paused', 'terminating' と (Real-Time Workshop に対して利用される)'external'を出力します。

目的 ダイナミックシステムのシミュレーション

表示 [t,x,y] = sim(model,timespan,options,ut);[t,x,y1, y2, ..., yn] = sim(model,timespan,options,ut);

詳細 sim コマンドは、Workspace I/O オプションを含めたすべてのシミュレーションパ ラメータダイアログの設定を使用して、Simulink モデルを実行します。

> 1番目の入力引数(モデル名)以外の入力引数は、空行列([])に設定できます。 sim コマンドは、空行列として指定された引数に対してはデフォルト値を適用し ます。sim コマンドの options 引数を利用して、オプションのシミュレーションパ ラメータを設定できます。この方法で指定されたパラメータによって、モデルで 指定されているパラメータが無効になります。

> 左側の引数を指定しない場合は、Simulation parameters ダイアログボックスの Workspace I/O パネル (5-18 ページの "Workspace I/O ページ")で指定されたシミュ レーションデータが記録されます。

> 連続システムをシミュレーションする場合、simset (5-43 ページの simset を参照) コマンドを使って solver パラメータを指定しなければいけません。純粋な離散モ デルに対しては、VariableStepDiscrete ソルバが適用されます。

- **引数** t 出力されるシミュレーションの時間ベクトル
 - × 出力されるシミュレーションの状態行列で、連続状態とそれに続く離散状態を含みます
 - y 出力されるシミュレーションの出力行列。各列にはポート番号順に、ルートレベルの Outport ブロックからの出力が含まれます。
 いずれかの Outport ブロックにベクトル入力がある場合、その出力は該当する数の列を必要とします
 - y1,...,yn 各 yi は、n 個のルートレベルの Outport ブロックを持つモデルに 対して関連する出力ポートの出力値です

model ブロック線図の名前

シミュレーションの開始時間と終了時間。つぎのいずれか1つを 指定します。
tFinal 終了時間を指定します。開始時間は0です。
[tStart tFinal] 開始時間と終了時間を指定します。
[tStart OutputTimes tFinal] 開始時間と終了時間、および t に出力され る時間列を指定します。一般に t は、より多くの時間列を含みま す。OutputTimes は、ダイアログボックスで Produce additional output を選択するのと等価です。
simset コマンドで作成される構造体として指定した、オプション のシミュレーションパラメータ (5-43 ページの simset を参照)。
トップレベルの Inport ブロックに対するオプションの外部入力。 ut は各時間ステップでの入力 u = UT(t) を指定する(文字列として 表現された)MATLAB 関数、もしくは、全ての入力ポートに対す る入力値 vs.時間のテーブル、もしくは、指定した各々のポート に関連するコンマで区切られたテーブルのリスト ut1, ut2,, で す。全てのポートに対するテーブル入力は、MATLAB 行列や構 造体です。個々のポートに対するテーブル入力は、構造体でなけ ればいけません。行列入力や構造体入力に関する詳細は、5-19 ページの"ベースワークスペースから入力の読み込み"を参照して ください。

つぎのコマンドは、Simulink モデル vdp を使って、Van der Pol の方程式をシミュ レーションします。ここでは、全てのパラメータはデフォルト値を利用します。

[t,x,y] = sim('vdp')

例題

つぎのコマンドは、Refine パラメータに対する値を定義し、その他は vdp モデル に関連付けられたパラメータ値を利用して Van der Pol 方程式をシミュレーショ ンします。

[t,x,y] = sim('vdp', [], simset('Refine',2));

つぎのコマンドは、1,000 秒間 Van der Pol 方程式をシミュレーションし、最後の 100 行の出力変数を保存します。シミュレーションは、t と y のみに対して値を 出力しますが、xFinal と呼ばれる変数に最終の状態ベクトルを保存します。 [t,x,y] = sim('vdp', 1000, simset('MaxRows', 100, 'OutputVariables', 'ty', 'FinalStateName', 'xFinal'));

参考

simset, simget

目的	figure ウィンドウにシミュレーションデータをプロット								
表示	simplot(data); simplot(time, data);								
詳細	simplot コマン をプロットし 見えます。出 けたり、印刷	^ν ドは、Handle Graphics figure ウィンドウにシミュレーションの出力 ます。プロットは、Scope ブロックのスクリーン上の表示のように 力を figure ウィンドウにプロットすることにより、出力に注釈を付 ができます。							
引数	data	出力ブロック (例:ルートレベルの Outport ブロックまたは To Workspace ブロック) で生成されるデータ、または Array, Structure, Structure with time ブロックで用いられる出力書式の データ (5-18 ページの "Workspace I/O ページ"を参照)。							
	time	シミュレーションの出力書式として Array または Structure を選 択したときに、出力ブロックが生成するサンプル時間のベクト ル。simplot コマンドは、データの書式が Structure with time であ る場合には、この引数を無視します。							

例題

つぎのコマンド

vdp set_param(gcs, 'SaveOutput', 'on') set_param(gcs, 'SaveFormat', 'StructureWithTime') sim(gcs) simplot(yout)

は、以下のように figure ウィンドウに vdp デモモデルの出力をプロットします。





sim, set_param

目的 sim コマンドのためのシミュレーションパラメータとソルバ特性を作成または編 集します。

表示 options = simset(property, value, ...); options = simset(old_opstruct, property, value, ...); options = simset(old_opstruct, new_opstruct); simset

詳細 simset コマンドは、指定されたシミュレーションパラメータおよびソルバプロパティが指定の値をもつ、options と呼ばれる構造体を作成します。未指定のパラメータと特性はすべて、それらのデフォルト値となります。パラメータまたは特性を一意的に識別するために十分な長さの先行文字を入力するだけで構いません。パラメータと特性に対しては大文字と小文字の区別は無視されます。

options = simset(property, value, ...) は、指定されたプロパティの値を設定し、options に構造体を保存します。

options = simset(old_opstruct, property, value, ...) は、既存の構造体である old_opstruct 内の指定されたプロパティを修正します。

options = simset(old_opstruct, new_opstruct) は、old_opstruct および new_opstruct の2つ の既存のオプション構造体を組み合わせて、options を作成します。new_opstruct に定義されるプロパティはすべて、old_opstruct に定義される同じプロパティを 変更します。

入力引数をもたない simset は、すべてのプロパティ名とその取り得る値を表示します。

get_param および set_param コマンドを用いて、これらの特性やパラメータの値を 取り出したり設定したりすることはできません。

パラメータ AbsTol 正のスカラ {1e-6}

絶対誤差許容値.このスカラは、状態ベクトルのすべての要素に適用されます。 AbsTolは、可変ステップソルバにのみ適用されます。

Decimation 正の整数 {1}

出力変数に対する間引き.出力変数 t, x, y に適用される間引きファクタ。間引き ファクタ1は、時間列に対するすべてのデータログを出力し、間引きファクタ2 は、1つおきのデータログを出力します。 DstWorkspace

base | {current} | parent

変数を割り当てるワークスペース. このプロパティは、出力変数あるいは To Workspace ブロック上の出力変数として定義した変数を割り当てるワークスペー スを指定します。

FinalStateName string {"}

最終状態変数名. このプロパティは、Simulink がシミュレーションの終了時にモ デルの状態を保存する変数の名前を指定します。

FixedStep positive scalar

固定ステップサイズ.このプロパティは、固定ステップソルバにのみ適用されます。モデルに離散系成分が含まれている場合、デフォルトは基本サンプル時間です。そうでない場合、デフォルトはシミュレーション区間の 50 分の 1 です。

InitialState vector {[]}

初期連続状態および初期離散状態.初期状態ベクトルは、(存在するなら)連続 状態と(存在するなら)それに続く離散状態から構成されます。モデル内で指定 される初期状態は InitialState で置き換えられます。デフォルトの空行列では、モ デル内で指定された初期状態値が使われます。

InitialStep

positive scalar {auto}

推奨初期ステップサイズ. このプロパティは、可変ステップソルバのみに適用されます。ソルバはまず、InitialStepのステップサイズを試みます。デフォルトではソルバが自動的に初期ステップサイズを決定します。

MaxOrder $1 | 2 | 3 | 4 | \{5\}$

ode15s の最高次数. This property applies only to ode15s.

MaxDataPoints nonnegative integer {0}

出力行の制限数. このプロパティは、t, x, y に出力される行数を、最後から MaxRows 点のデータログに制限します。デフォルトの0を指定すると、制限は 適用されません。

MaxStep

positive scalar {auto}

ステップサイズの上限.このプロパティは、可変ステップソルバのみに適用され、シミュレーション区間の 50 分の 1 にデフォルト設定されます。

OutputPoints {specified} | all

出力点の決定. specified に設定すると、ソルバは timespan で指定した時間でのみ、 出力 t, x, y を生成します。all に設定すると、t, x, y には、ソルバが使った時間ス テップも含まれます。 OutputVariables

 $\{txy\} \mid tx \mid ty \mid xy \mid t \mid x \mid y$

出力変数の設定. プロパティ文字列に ⁱť, ⁱx', ⁱy' がないと、ソルバは対応する出力 t, x, y に空行列を生成します。

Refine

positive integer {1}

出カリファインファクタ.このプロパティは、指定したファクタだけ出力点数を 増加させ、より滑らかな出力を生成します。Refine は可変ステップソルバのみに 適用されます。出力時間を指定すると、Refine は無視されます。

RelTol positive scalar {1e-3}

相対誤差許容値.このプロパティは、状態ベクトルのすべての要素に適用されます。各積分ステップの推定誤差は、以下を満足します。

e(i) <= max(RelTol*abs(x(i)),AbsTol(i))

このプロパティは、可変ステップソルバのみに適用され、0.1%の精度に対応する 1e-3 にデフォルト設定されます。

Solver

VariableStepDiscrete | ode45 | ode23 | ode113 | ode15s | ode23s | FixedStepDiscrete | ode5 | ode4 | ode3 | ode2 | ode1

時間進行方法.このプロパティは、時間を進めるためにどのソルバを使用するか を指定します。

SrcWorkspace {base} | current | parent

式を評価するワークスペース.このプロパティは、モデル内で定義されている MATLAB表現を評価するワークスペースを指定します。

Trace

'minstep', 'siminfo', 'compile' { "}

トレース機能.このプロパティは、シミュレーショントレース機能を有効にします(コンマで区切ったリストとして1つまたは複数を指定します)。

- 'minstep':トレースフラグは、解の変化が余りにも突然で可変ステップソルバのステップサイズで誤差許容値を満足させることができないような場合に、シミュレーションを停止するように指定します。デフォルトでは、Simulinkがワーニングメッセージを出力してシミュレーションを続行します。
- 'siminfo': トレースフラグは、シミュレーションの開始時に有効なシミュレーションパラメータの簡単な要約を提供します。
- 'compile': トレースフラグは、ブロック線図モデルのコンパイル段階を表示します。

ZeroCross {on} | off **ゼロクロッシング検出の有効/無効**:このプロパティは、可変ステップソルバの みに適用されます。offに設定すると、可変ステップソルバは、固有のゼロク ロッシング検出をもつブロックに対するゼロクロッシングを検出しません。ソル バは、誤差許容値を満足させるためだけにそれらのステップサイズを調整しま す。

つぎのコマンドは、MaxRows と Refine のパラメータに対して値を定義する、 myopts と呼ばれるオプション構造体を生成します。他のパラメータにはデフォ ルト値を使用します。

myopts = simset('MaxDataPoints', 100, 'Refine', 2);

つぎのコマンドは、vdp model モデルを 10 秒間シミュレーションし、myopts に定 義されているパラメータを使用します。

[t,x,y] = sim('vdp', 10, myopts);

sim, simget

参考

例題

目的 オプション構造体のプロパティとパラメータを取り出します。

表示 struct = simget(model)

value = simget(model, property)

value = simget(OptionStructure, property)

詳細 simget コマンドは、指定した Simulink モデルに対するシミュレーションパラ メータとソルバのプロパティ値を取り出します。変数名を用いてパラメータまた はプロパティを定義すると、simget は変数の名前ではなく変数の値を出力しま す。変数がワークスペースに存在しない場合には、エラーメッセージが表示され ます。

> struct = simget(model) は、指定した Simulink モデルに対する現在のオプション構造体を出力します。オプション構造体は、sim コマンドと simset コマンドを用いて 定義されます。

> value = simget(model, property) は、指定されたシミュレーションパラメータまたはソ ルバのプロパティ値をモデルから抽出します。

> value = simget(OptionStructure, property)は、指定されたシミュレーションパラメータ またはソルバのプロパティ値をOptionStructureから抽出し、構造体に値が指定さ れていない場合には空行列を出力します。propertyは、興味のあるパラメータと プロパティ名のリストを含むセル配列とすることができます。セル配列を使用す ると、出力もセル配列です。

> プロパティ名は、一意的に識別するために必要な先行するキャラクタを入力する だけで構いません。プロパティ名の場合、大文字と小文字の区別は無視されま す。

例題

つぎのコマンドは、vdp モデルに対するオプション構造体を取り出します。

options = simget('vdp');

つぎのコマンドは、vdp モデルに対する Refine プロパティの値を取り出します。

refine = simget('vdp', 'Refine');

参考 sim, simset

シミュレーション結果の解 析

出力軌 Scope 出力変 To Wo	跡の ブロ 数の rksp)表 ーッ D何 ace	示ク 用 つ	の 引注 ブロ	・ 使 い い	・ 用 ・	・ 法 フロ	י ד ת	· · ·	• · ·	· · 法	• • •	• • •	•	•	• • •	•	•	• • •	6-2 6-2 6-2 6-3								
線形化			•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•		•	•		•	•	•	6-4
平衡点	の決	定		•	•	•	•	•	•	•		•	•	•	•	•	•	•		•		•	•		•	•	•	6-7
linfun			•	•	•	•	•	•	•	•		•	•	•	•	•	•	•	•	•	•	•	•		•	•		6-9
trim .			•	•			•	•	•	•		•	•	•	•	•		•					•	•	•		.(6-12

出力軌跡の表示

Simulink からの出力軌跡は、つぎの3つの手法のいずれかを用いてプロットする ことができます。

- ・ 信号を Scope ブロックまたは XY Graph ブロックに入力する。
- 出力を出力変数に書き出し、MATLAB のプロットコマンドを使用する。
- To Workspace ブロックを用いて出力をワークスペースに書き出し、MATLAB のプロットコマンドを用いて結果をプロットする。

Scope ブロックの使用法

シミュレーション中、Scope ブロック上に出力軌跡を表示することができます。 つぎの簡単なモデルは、Scope ブロックの使い方の例を示したものです。



Scope 上の表示は出力軌跡を示します。Scope ブロックでは、興味のある領域を 拡大したり、ワークスペースにデータを保存したりすることができます。

XY Graph ブロックでは、ある信号を別の信号に対比させてプロットすることができます。

これらのブロックについては、第9章の"ブロックリファレンス"で説明します。

出力変数の使用法

時間履歴と出力履歴をワークスペースに出力ことによって、MATLABのプロットコマンドで出力軌跡を表示し、それに注釈を付けることができます。



Out のラベルの付いたブロックは、Signals & Systems ライブラリにある Outport ブロックです。出力軌跡 yout は、積分ソルバによりワークスペースに出力され ます。詳細については、5-18 ページの "Workspace I/O ページ"を参照してください。

このシミュレーションは、Simulation Parameters ダイアログボックスの Workspace I/O ページ上で時間、出力、状態に対する変数を指定することによっ て、Simulation メニューから実行することもできます。それから以下を用いてこれらの結果をプロットすることができます。

plot(tout,yout)

To Workspace ブロックの使用法

To Workspace ブロックを使用して、出力軌跡を MATLAB ワークスペースに出力 することができます。つぎのモデルは、この使い方を示したものです。



変数 y と t は、シミュレーションが完了するとワークスペースに現れます。時間 ベクトルは、Clock ブロックを To Workspace ブロックに結線することによって保 存されます。時間ベクトルは、メニュー方式のシミュレーションの場合、 Simulation Parameters ダイアログボックスの Workspace I/O ページ上で時間に対 する変数名を入力するか、sim コマンドを用いてそれを出力することによっても 得られます (詳細については、5-18 ページの "Workspace I/O ページ"を参照して ください)。

To Workspace ブロックは、ベクトル入力を受け入れ、各入力要素の軌跡は結果 として得られるワークスペース変数の列ベクトルとして保存されます。

線形化

Simulink は、状態空間行列 A, B, C, D の形式で線形モデルを抽出する、関数 linmod と dlinmod を提供します。状態空間行列は、つぎのような線形の入出力関 係を記述します。

- $\dot{x} = Ax + Bu$
- y = Cx + Du

ここで、x, u, yはそれぞれ状態、入力、出力ベクトルです。たとえば、つぎのモ デルを lmod とすると、



この Simulink システムの線形モデルを抽出するには、つぎのコマンドを入力します。

[A,B,C,D] = linmod('lmod')A =-2 -1 -1 1 0 0 0 1 -1 B = 1 0 0 C =0 1 0 0 0 -1 D = 0 1

入力と出力は、Signals & Systems ライブラリにある Inport ブロックと Outport ブ ロックを用いて定義しなければなりません。Sources ライブラリのブロックと Sinks ライブラリのブロックは、入力および出力としては機能しません。Inport ブロックは、Sum ブロックを用いて Sources のブロックと併用することができま す。データが状態空間型であるか、または LTI オブジェクトに変換されると、 Control System Toolbox 内の関数を適用して、さらに解析を続けることができま す。

- LTI オブジェクトへの変換: sys = ss(A,B,C,D);
- ボード周波数プロット:
 bode(A,B,C,D) or bode(sys)
- 線形化した時間応答: step(A,B,C,D) or step(sys) impulse(A,B,C,D) or impulse(sys) lsim(A,B,C,D,u,t) or lsim(sys,u,t)

Control System Toolbox と Robust Control Toolbox 内の他の関数は、線形制御システム設計のために使用することができます。

モデルが非線形である場合、線形化されたモデルを抽出する操作点を選択することができます。非線形モデルは、モデルを抽出する摂動のサイズに対しても敏感です。これらは、打ち切り誤差と丸め誤差間の調整を図るように選択しなければなりません。linmod に対するオプションの引数で、操作点と摂動点を指定します。

[A,B,C,D] = linmod('sys', x, u, pert, xpert, upert)

離散システム、または連続系と離散系の混成システムの場合、線形化のために関数 dlinmod を使用します。これは、右辺の2番目の引数で線形化を実行するサン プル時間を指定しなければならないということを除いて、linmod と同じ呼び出 し構文をもっています。詳細については、6-9ページの linfun コマンドの説明を参 照してください。

Derivative または Transport Delay ブロックを含むモデルを線形化するために linmod を使用するのはトラブルのもとです。線形化の前に、これらのブロック を、問題を回避する特別に設計されたブロックで置き換えてください。これらの ブロックは、Simulink Extras サブライブラリ内の Linearization ライブラリにあり ます。Extras ライブラリには、Blocksets & Toolboxes アイコンを開いてアクセス します。

- Derivative ブロックの場合、Switched derivative for linearization を用います。
- Transport Delay ブロックの場合、Switched transport delay for linearization を使います(このブロックを使用するには、Control System Toolbox が必要です)。

Derivative ブロックを使用する場合は、他のブロックに導関数の項を組み入れる ように試みることもできます。たとえば、Transfer Fcn ブロックと直列の Derivative ブロックがある場合、つぎの形式の単一の Transfer Fcn ブロックを用 いた方がより実現しやすくなります(ただし、これが常に可能であるとは限りま せん)。

 $\frac{s}{s+a}$

この例では、図の左側のブロックは右側のブロックで置き換えることができます。



平衡点の決定

Simulinkの関数 trim は、定常状態の平衡点を決定します。たとえば、Imod というモデルを検討します。



関数 trim は、両方の出力を1に設定する入力と状態の値を見つけるために使用 することができます。まず、状態変数(x)と入力値(u)に対して初期推定を行い、 出力(y)に対する希望値を設定します。

x = [0; 0; 0];u = 0;y = [1; 1];

どの変数が固定で、どの変数が可変であるかを示すために、インデックス変数を 使用します。

ix = []; % Don't fix any of the states
iu = []; % Don't fix the input
iy = [1;2]; % Fix both output 1 and output 2

trim を呼び出すと解が出力されます。丸め誤差のため結果は異なる場合があります。

[x,u,y,dx] = trim('lmod',x,u,y,ix,iu,iy)

x = 0.0000 1.0000 u = 2 y = 1.0000 1.0000 dx = 1.0e-015 * -0.2220 -0.0227 0.3331

平衡点の問題には解が存在しない場合があることに注意してください。その場合 には、trim はまず導関数をゼロに設定するよう試みた後で、希望する結果からの 最大偏差が最小となるような解を出力します。trim の構文の説明については、 6-12 ページのtrim を参照してください。

目的 操作点周辺でシステムの線形状態空間モデルを抽出します。

表示
[A,B,C,D] = *linfun*('sys', x, u)
[num,den] = *linfun*('sys', x, u)
sys_struc = *linfun*('sys', x, u)

詳細

引数 *linfun* sys x と u *linmod*、*dlinmod* または *linmod*2 線形モデルを抽出する *Simulink* システムの名前 太 と u 状態ベクトルと入力ベクトル。これを指定すると、線形モデルを

* 2 u 私感へりドルとハガベクドル。これを指定すると、緑形モナルを 抽出する操作点が設定されます。

linmod は、Simulink モデルとして記述される常微分方程式のシステムから線形モデルを取り出します。linmod は、線形化された入出力関係を記述する、状態空間型 A, B, C, D の線形モデルを出力します。

 $\dot{x} = Ax + Bu$ y = Cx + Du

入力と出力は、Inport ブロックと Outport ブロックを用いて、Simulink ブロック 線図で示されます。

[A,B,C,D] = linmod('sys', x, u) は、状態変数 x と入力 u で指定した操作点周辺の sys の 線形化モデルを取り出します。x と u を省略した場合、デフォルト値はゼロで す。

[num,den] = *linfun*('sys', x, u) は、伝達関数の形で線形化モデルを戻します。

sys_struc = *linfun*('sys', x, u) は、状態名、入力と出力名、および操作点の情報を含む 線形化モデルの構造体を戻します。

離散時間システムの線形化

関数 dlinmod は、任意のサンプリング時間で、離散システム、マルチレートシス テム、連続と離散のハイブリッドシステムを線形化することができます。 dlinmod に対しては linmod に対するのと同じ呼び出し構文を使用しますが、2 番 目の引数として線形化を実行するサンプル時間を挿入します。たとえば、

[Ad,Bd,Cd,Dd] = dlinmod('sys', Ts, x, u);

は、サンプル時間が Ts で、状態ベクトル x と入力ベクトル u で与えられる操作 点での離散状態空間モデルを生成します。離散システムの連続モデル近似を得る ためには、Ts を 0 に設定します。

線形、マルチレート、離散、および連続ブロックから構成されるシステムの場合、dlinmod は、つぎの条件の場合に、変換したサンプリング時間 Ts で同一の 周波数と時間応答(一定入力に対する)をもつ線形モデルを生成します。

• Ts はシステム内のすべてのサンプリング時間の整数倍である。

システムは安定している。

最初の条件を満足しないシステムでは、一般的に、線形化はある時間-変動シス テムです。これは dlinmod が戻す [A,B,C,D] 状態空間モデルで表現できません。

線形化した行列 Ad の固有値を計算すると、システムの安定性が示されます。 Ts>0 で、固有値がつぎのステートメントで決まるように単位円内にある場合、 システムは安定です。

all(abs(eig(Ad))) < 1

同様に、Ts=0で、固有値がつぎのステートメントで決まるように左半面にある 場合、システムは安定です。

all(real(eig(Ad))) < 0

システムが不安定でサンプル時間が他のサンプル時間の整数倍でない場合、 dlinmod は Ad と Bd 行列を生成し、これらは複素数かもしれません。しかし、こ のような場合でも Ad 行列の固有値は安定性をよく示しています。

dlinmod を用いてシステムのサンプル時間を他の値に変換したり、線形離散シス テムを連続システムに変換したり、あるいはその逆を行ったりすることができま す。

連続システムまたは離散システムの周波数応答は、bode コマンドを使用することによって求めることができます。

デフォルトでは、システム時間はゼロに設定されています。時間に依存するシス テムの場合、変数 pert を 2 要素ベクトルに設定することができます。ここで、2 番目の要素は、線形モデルを得る t の値を設定するために使用します。

非線形モデルから線形モデルへの状態の順序は維持されます。Simulink システム の場合、各状態に関連したブロック名を含む文字列変数は、以下を用いて得るこ とができます。

注意

[sizes,x0,xstring] = sys

ここで、xstring は i 番目の行が i 番目の状態に関連したブロック名である文字列のベクトルです。入力と出力には、ブロック線図で順番に番号が付けられます。

単入力多出力システムの場合、ルーチン ss2tf を用いて伝達関数型に変換したり、 ss2zp を用いて零点 - 極型に変換することができます。また、線形化されたモデ ルを ss を用いて LTI オブジェクトに変換することもできます。この関数は、状 態空間型の LTI オブジェクトを生成し、LTI オブジェクトは tf または zpk を用い てさらに伝達関数型や零点 - 極 - ゲイン型に変換することができます。

Derivative または Transport Delay ブロックを含むモデルの線形化は、問題を含んでいます。詳細については、6-4 ページの "線形化"を参照してください。

目的 動的システムの平衡点を求めます。

表示

[x,u,y,dx] = trim('sys')

[x,u,y,dx] = trim('sys',x0,u0,y0)[x,u,y,dx] = trim('sys',x0,u0,y0,ix,iu,iy)

[x,u,y,dx] = trim(sys,x0,u0,y0,ix,iu,iy,dx0,idx)

[x,u,y,dx] = trim('sys',x0,u0,y0,ix,iu,iy,dx0,idx,options)

[x,u,y,dx] = trim('sys',x0,u0,y0,ix,iu,iy,dx0,idx,options,t)

[x,u,y,dx,options] = trim('sys',...)

詳細

つりあいの位置として知られる trim ポイントは、動的システムが定常状態とな るパラメータ空間のある点です。たとえば、航空機の平衡点は、航空機が水平に 直進するための制御点です。数値的には、平衡点はシステムの状態微係数がゼロ となる点です。trim は、初期値から始めてもっとも近い平衡点を探し出すまで、 逐次二次計画アルゴリズムを使って探索します。陽的に、または陰的に初期値を 与える必要があります。trim は平衡点が求まらないとき、探索した値を出力しま す。ここで、状態微係数は min-max の観点で、ゼロにもっとも近くなります。 つまり、微係数のゼロからの最大偏差を最小化する点を出力します。trim は、指 定した入力、出力、状態の条件に一致する平衡点を求めることができます。ま た、システムが指定した方法で変化する点を求めることをできます。つまり、シ ステムの状態微係数が指定した非ゼロの値と等しくなる点を求めることができま す。

[x,u,y] = trim('sys') は、システムの初期状態 x0 にもっとも近いつりあいの位置を求めます。厳密には trim は [x-x0,u,y] の最大絶対値を最小化するつりあいの位置を求めます。trim はシステムの初期状態に近いつりあいの位置が求まらないとき、システムがもっともつりあう位置を出力します。厳密には abs(dx-0) を最小化する点を出力します。つぎのコマンドを使って x0 を得ることができます。

[sizes,x0,xstr] = sys([],[],[],0)

[x,u,y] = trim('sys',x0,u0,y0) は、x0,u0,y0 にもっとも近い平衡点を求めます。つまり、 abs([x-x0; u-u0; y-y0]) の最大値を最小化する点を求めます。

コマンド

trim('sys', x0, u0, y0, ix, iu, iy)

は、指定した状態、入力、出力の条件を満足する x0, u0, y0 にもっとも近い平衡 点を求めます。整数ベクトル ix, iu, iy は、満足すべき x0, u0, y0 の値を選択しま す。trimは、指定した条件を正確に満足する平衡点を求められないとき、つぎの条件を満足するようなもっとも近い値を出力します。

abs([x(ix)-x0(ix); u(iu)-u0(iu); y(iy)-y0(iy)])

指定した非平衡点を求めるためには、つぎの記法を利用します。

[x,u,y,dx] = trim('sys', x0, u0, y0, ix, iu, iy, dx0, idx)

つまり、システムの状態微係数が、ある指定した非ゼロの値となる点を求めま す。ここで、dx0 は探索の開始点での状態微係数を指定し、idx はその探索が満 足すべき dx0 の値を選択します。

オプションの引数 options は、trim 平衡点を求めるために利用する最適化関数に 受け渡す最適化パラメータの配列です。最適化関数は、最適化の手順を制御する ために、また過程の情報を出力するためにこの配列を利用します。trim は探索過 程の最後の段階で、options 配列を出力します。この方法で潜在的な最適化の過 程を公開することにより、trim は平衡点探索のモニタリングとファインチューニ ングができます。

最適化のための配列の要素のうち、平衡点を求めるために特に有効なものが5つ あります。つぎの表は、各要素が平衡点探索にどのように影響するかを記述して います。

No.	デフォル ト	詳細
1	0	表示オプションを指定します。0 は何も表示しません。1 は表形式で出力します。-1 はワーニングメッセージを出 力しません。
2	0.0001	探索を終了するために達成されなければならない求めら れた平衡点の精度
3	0.0001	探索を終了するために達成されなければならない目的関 数の精度
4	0.0001	探索を終了するために達成されなければならない状態微 係数の精度
10	N/A	平衡点探索を実行する繰り返し回数を出力します。

options 配列の詳細については、*Optimization Toolbox User's Guide* を参照してくだ さい。

例題

線形状態空間モデルについて考えます。

 $\dot{x} = Ax + Bu$

y = Cx + Du

A, B, C, D 行列は、つぎのような sys と呼ばれるシステムです。

例題1

平衡点を求めるには、以下を使用します。

[x,u,y,dx,options] = trim('sys')

 $x = 0 \\
 0 \\
 u = 0 \\
 y = 0 \\
 0 \\
 dx = 0 \\
 0 \\
 0$

必要な反復回数は、つぎのとおりです。 options(10) ans = 7 x = [1;1], u = [1;1] の近くの平衡点を求めるには、以下を入力します。 x0 = [1:1]:

x0 = [1;1]; u0 = [1;1]; [x,u,y,dx,options] = trim('sys', x0, u0);

 $\begin{array}{l} x = \\ 1.0e - 11 * \\ -0.1167 \\ u = \\ 0.3333 \\ 0.0000 \\ y = \\ -1.0000 \\ 0.3333 \\ dx = \\ 1.0e - 11 * \\ 0.4214 \\ 0.0003 \end{array}$

必要な反復回数は、つぎのとおりです。

options(10) ans = 25

例題3

例題2

出力を1に固定した平衡点を求めるには、以下を使用します。

y = [1;1]; iy = [1;2]; [x,u,y,dx] = trim('sys', [], [], y, [], [], iy) x = 0.0009 -0.3075 u =

-0.5383 0.0004 y = 1.0000 1.0000 dx =1.0e-16 * -0.0173 0.2396 出力を1に固定し、微係数を0と1に設定した平衡点を求めるには、以下を使用 します。 y = [1;1];iy = [1;2]; dx = [0;1];idx = [1;2];[x,u,y,dx,options] = trim('sys',[],[],y,[],[],iy,dx,idx) x = 0.9752 -0.0827 u = -0.3884 -0.0124 **y** = 1.0000 1.0000 dx =0.0000 1.0000 必要な反復回数は、つぎのとおりです。 options(10) ans =13

例題4

制限

trim は与えられた初期値から始めると、求まる平衡点は局所的値のみです。他のより適切な平衡点が存在するかもしれません。従って、特殊な応用のために更に
適切な平衡点を求めたい場合、x,u,yに対する多くの初期推定値を試してみることが重要です。

アルゴリズム trim は、平衡点を求めるために逐次二次計画アルゴリズムを用います。アルゴリズムの詳細については、*Optimization Toolbox User's Guide* をご参照ください。

マスクを使ったブロックの カスタマイズ

はじめに	2
マスクされたサンプルサプシステム	;
マスクタイアログボックスのフロンフトの作成7-4	ŀ
ブロック記述とヘルブテキストの作成	5
ブロックアイコンの作成 7-6)
Mask Editor: 概要	;
Initialization $\checkmark - \vartheta$)
プロンプトと関連する変数 7-9)
マスクされたブロックパラメータに対するデフォルト値 7-13	3
変更可能なパラメータ 7-13	3
変更可能なパラメータ 7-13	5
Icon ページ	,
ブロックアイコン上でのテキストの表示 7-17	1
ブロックアイコン上でのグラフィックスの表示 7-19)
マスク上でのイメージの表示)
ブロックアイコン上での伝達関数の表示 7-21	L
アイコンプロパティの制御 7-22	2
Documentation $\checkmark - ?$	5
Mask Type $77 - \mu$ F	5
Block Description $74 - \mu F$ 7-26	5
Mask Help Text $7 - \mu F$	7
Self-Modifying マスクプロックの作成	3
マスクブロックに対する動的ダイアログの作成)
マスクブロックダイアログのパラメータの設定 7-29)
定義済みマスクダイアログパラメータ 7-30)

はじめに

マスキングは、サブシステム用のダイアログボックスとアイコンをカスタマイズ できる、強力な Simulink の機能です。マスキングを使用すると、つぎのことが 可能になります。

- サブシステムの多くのダイアログボックスを1つのダイアログボックスで置き 換えることによって、モデルの使用を単純化します。モデルの各ブロックを 開いたパラメータ値を入力する代わりに、それらのパラメータ値は、マスク したダイアログボックス上で入力し、マスクされているサブシステムのブ ロックに渡すことができます。
- ユーザ独自のブロック説明、パラメータフィールドのラベル、ヘルプテキストをもつダイアログボックスを定義することによって、より詳しい説明と有効なユーザインタフェースを提供します。
- 値がブロックパラメータによって異なる変数を計算するコマンドを定義する ことができます。
- ブロックの目的を記述するブロックアイコンを作成することができます。
- サブシステムの内容をカスタマイズしたインタフェースで隠すことによって、
 不用意なサブシステムの変更を回避することができます。
- 動的ダイアログの作成

マスクされたサンプルサブシステム

この簡単なサブシステムは、直線 y = mx + b の方程式をモデル化しています。



通常、Subsystem ブロックをダブルクリックすると、Subsystem ブロックが開き、 別のウィンドウにそのブロックを表示します。mx + b のサブシステムには、Gain パラメータを m として指定する Slope と名づけられた Gain ブロックと、定数値 パラメータを b として指定する Intercept と名づけられた Constant ブロックが含ま れています。これらのパラメータは直線の勾配と切片を表します。

この例では、サブシステムのためにダイアログボックスとアイコンをカスタマイ ズします。このダイアログボックスには、勾配と切片に対するプロンプトが含ま れます。マスクの作成後、Subsystem ブロックをダブルクリックすると、マスク ダイアログボックスが開きます。マスクダイアログボックスとアイコンはつぎの ようになります。

Block Parameters: mx + b	マスクダイアログボックス
SampleMaskedBlock (mask)	
Models the equation for a line, $y = mx + b$. The slope and intercept are mask block parameters.	
Parameters	
Slope:	
3	ブロックマイコン
Intercept:	フロックアイコン
2	
OK Cancel Help Apply	*/>

このサブシステムでは、Slope と Intercept の値をマスクダイアログボックスに入 力します。Simulink は、基礎になっているサブシステム内のブロックがこれらの 値を利用できるようにします。このようにサブシステムをマスクすると、アプリ ケーション固有のパラメータである Slope と Intercept をもつ、自己完結型関数ユ ニットが作成されます。マスクは、これらのマスクパラメータを基礎ブロック の一般パラメータに渡します。複雑なサブシステムでも、組み込みの Simulink ブロック同様の外観と操作感覚をもつ新しいインタフェースによって簡約化され ます。 このサブシステムに対してマスクを作成するには、つぎの作業を行います。

- マスクダイアログボックスのパラメータに対するプロンプトを指定します。
 この例では、マスクダイアログボックスに勾配と切片に対するプロンプトが 含まれます。
- 各パラメータの値を格納するために使用する変数名を指定します。
- ブロックの説明とヘルプテキストからなるブロックのドキュメントを入力します。
- ブロックアイコンを作成するための描画コマンドを指定します。
- ・ 描画コマンドにとって必要な変数を提供するコマンドを指定します(この例では1つもありません)。

マスクダイアログボックスのプロンプトの作成

このサブシステムに対するマスクを作成するには、Subsystem ブロックを選択し、Edit メニューから Mask Subsystem を選択します。

この節のはじめに示したマスクダイアログボックスは、主に Mask Editor の Initialization ページ上で作成されます。このサンプルモデルの場合、ページはつ ぎのとおりです。

🚮 Mask Edito	or: untitled/mx + b		_ 🗆 ×	<	
Icon	Initialization Docume	entation		1	
Mask type:	SampleMaskedBlock				
	Prompt	Туре	Variable		
Add	Slope:		edit m 📥		
Delete	Intercept: < <end of="" paramete<="" td=""><td>r list>></td><td>edit 📣</td><td></td><td>が八刀した値を休持するフロンフト、 タイプ、変数。</td></end>	r list>>	edit 📣		が八刀した値を休持するフロンフト、 タイプ、変数。
	1				
Down			▼ ▶		
Prompt: Sl	ope:	Control type:	Edit 💌 🖷		
Variable: m		Assignment:	Evaluate 💌 🗸	✐	。パラメータフィールドの値を入力した り始集を実行
Popup strings:				1	り補朱で天口。
Initialization co	ommands:				
		•			 アイコン描画コマンドまたはマスクされているサブシステムのブロックが使用する変数を定義するコマンド群。
OK	Cancel Unma	sk. Help	Apply		

Mask Editor では、マスクパラメータのつぎの属性を指定することができます。

- プロンプト パラメータを説明するテキストラベル。
- コントロールタイプ パラメータ値を入力したり選択したりする方法を決める ユーザインタフェースコントロールのスタイル。
- 変数 パラメータ値を格納する変数名。

一般に、マスクされたパラメータはそれらのプロンプトで参照する方が便利で す。この例では、勾配に関連したパラメータを Slope パラメータ、切片と関連し たパラメータを Intercept パラメータとして参照します。

勾配と切片は、エディットテキストとして定義されます。この場合、ユーザはマ スクダイアログボックスのエディットフィールドに値を入力します。これらの値 は、マスクワークスペース(7-14 ページの"マスクワークスペース"を参照)内の 変数に格納されます。マスクされたブロックは、マスクワークスペース内の変数 のみにアクセスすることができますこの例では、勾配に対して入力した値は変 数mに割り当てられます。マスクされたサブシステム内の Slope ブロックは、勾 配パラメータに対する値をマスクワークスペースから取り出します。つぎの図 は、Mask Editor 内の勾配パラメータの定義が、実際のマスクダイアログボック スパラメータにどのように反映されるかを示しています。

			Block Parameters: mx + b	×
			SampleMaskedBlock (mask)	
			Models the equation for a line, $y = mx + b$. The slope and intercept are mask block parameters.	
			Parameters	
			Slope:	
			3	
			Intercept:	
Prompt Type	Variable		2	
Slope: 🗖	edit m	. /	1	
Intercept:	edit b	r	OK Cancel Help Apply	
< <end list="" of="" parameter="">></end>				

勾配と切片に対するマスクパラメータを作成したら、OK ボタンを押します。それから、Subsystem ブロックをダブルクリックして、新しく構築されたダイアロ グボックスをオープンします。Slope パラメータには3を入力し、Intercept パラ メータには2を入力します。

ブロック記述とヘルプテキストの作成

マスクタイプ、ブロック説明、ヘルプテキストは、Documentation ページで定義 します。このサンプルのマスクブロックの場合、ページは、つぎのとおりです。

Mask Editor: untitled/mx + b	
Block description: Models the equation for a line, y = mx + b. The slope and intercept are mask block parameters.	Block Parameters: mx + b SampleMaskedBlock (mask) Models the equation for a line, y = mx + b. The slope and intercept are mask block parameters. Parameters Slope: 3
Block help: Enter the slope (m) and intercept (b) in the block dialog parameter fields. The block generates y for a given input x.	Intercept ☐ Cancel → Help ,
OK Cancel Unmask Help Apply	

ブロックアイコンの作成

これまで、mx + b サブシステムのためにダイアログボックスをカスタマイズして きました。しかし、Subsystem ブロックは依然として一般の Simulink サブシステ ムアイコンを表示しています。このマスクされたブロックに対して、直線の勾配 を示すプロットがアイコンとして適切です。勾配が3の場合、そのアイコンはつ ぎのようになります。



ブロックのアイコンは、Icon ページ上で定義されます。このブロックの場合、 Icon ページは、つぎのようになります。

🛃 Mask Editor: slopeintercept/mx +	b		
Icon Initialization Docum	entation		
Mask type: SampleMaskedBlock			
Drawing commands:			
plot([0 1],[0 m] + (m<0))	•		描写コマンド
	Icon frame: Visible	▼	
lco	n transparency: Opaque		> アイコンプロパティ
	Icon rotation: Fixed	•	
Drawi	ng coordinates: Normalized		
OK Cancel Unma	isk Help Aj		

描画コマンドは、(0,0)から(0,m)までの直線をプロットします。勾配が負の場合には、直線が1だけ上にシフトして、プロックの表示描画領域内に保持されます。

描画コマンドは、マスクワークスペース内のすべての変数にアクセスすることが できます。異なる勾配の値を入力すると、アイコンはプロットされる直線の勾配 を自動的に更新します。

アイコンのプロパティのリストの最下部にある Drawing coordinates パラメータ として Normalized を選択すると、アイコンは左下隅が (0,0) で右上隅が (1,1) の フレームに描かれます。このパラメータについては本章の後半で説明します。詳 細は、7-19 ページの"プロックアイコン上でのグラフィックスの表示"を参照し てください。

Mask Editor: 概要

サブシステムをマスクするには (Subsystem ブロックのみマスクできます)、 Subsystem ブロックを選択した後で Edit メニューから Mask Subsystem を選択し ます。Mask Editor が表示されます。Mask Editor は 3 ページから構成されており、 それぞれのページはマスクの異なった側面を扱っています。

- Initialization ページでは、マスクダイアログボックスパラメータのプロンプトの定義と説明を行い、パラメータと関連する変数に名前を付け、初期化コマンドを指定することができます。
- Icon ページでは、ブロックアイコンを定義することができます。
- Documentation ページでは、マスクタイプを定義し、ブロックの説明とヘルプ を指定することができます。

Mask Editor の最下部に、つぎの5つのボタンが表示されます。

- OK ボタンは、すべてのページにマスク設定値を適用し、Mask Editor を閉じます。
- Cancel ボタンは、変更を適用せずに Mask Editor を終了します。従って、最後に Apply ボタンを押した設定のままです。
- Unmask ボタンは、マスクを解除して Mask Editor を閉じます。マスクを再度ア クティブにできるように、マスク情報は保存されます。マスクを再度アク ティブにするには、ブロックを選択し、Create Mask を選択します。Mask Editor が開き、以前の設定値を表示します。アクティブでないマスク情報は、モデ ルを閉じると破棄され、復帰することはできません。
- Help ボタンは、本章の内容を表示します。
- Apply ボタンは、すべてのマスキングページに表示される情報を用いて、マス クを作成および変更します。Mask Editor は開かれたままです。

マスクを解除しないでマスクの基になるシステムを見るためには、Subsystem ブ ロックを選択してから、Edit メニューの Look Under Mask を選択します。ブ ロックがサブシステムでない場合、このコマンドは基礎となっているブロックの ダイアログボックスを開きます。プロックのマスクは影響を受けません。

Initialization ページ

マスクインタフェースでは、マスクされたシステム内のブロックに対するパラ メータ値を入力することができます。マスクインタフェースは、Initialization ページ上で、パラメータ値に対するプロンプトを定義することによって作成しま す。mx+b のマスクされたサンプルシステムに対する Initialization ページは、つ ぎのように表示されます。

March Call	and the state of the state of the			1	
Mask Eult	or: undded/llix + D			4	
Icon	Initialization Do	cumentation			
Mask type:	SampleMaskedBloc	k			
	Prompt	Туре	Variable		
Add	Slope:		edit m 📥	<	プロンプトのリスト
Delete	Intercept:	eter list>>	edit b		
11-					
Up	_		T		
Down					
Prompt: S	lope:	Control type:	Edit 💌		各パラメータプロンプトの完全
Variable:	I	Assignment:	Evaluate 💌 🔺		な記述
Popup strings	5			Í l	
Initialization c	ommands:				
			•	_	初期化コマンド
OK	Cancel L	Inmask Help	Apply		

プロンプトと関連する変数

プロンプトは、ブロックパラメータに対する値をユーザが入力したり選択したり する際に役立つ情報を提供します。プロンプトは、Prompt リストで表示される 順序に従って、マスクダイアログボックスに表示されます。

プロンプトを定義する場合には、同時にパラメータ値を格納する変数を指定し、 プロンプトに対するコンロトールのスタイルを選択し、変数に値がどのように格 納されるかを示します。

Assignment タイプが Evaluate の場合、ユーザが入力する値は、それを変数に割 り当てる前に MATLAB が評価します。タイプが Literal の場合、ユーザが入力 する値は評価されませんが、文字列として変数に割り当てられます。 たとえば、ユーザが文字列 gain をエディットフィールドに入力し、Assignment タ イプが Evaluate の場合、文字列 gain は MATLAB によって評価され、結果は変数 に割り当てられます。タイプが Literal の場合、文字列は MATLAB によって評価 価されないので、変数には文字列 'gain' が含まれています。

入力した文字列と評価値のいずれもが必要な場合には、Literalを選択します。 それから、初期化コマンドで MATLAB の eval コマンドを使います。たとえば、 LitVal が文字列 'gain'の場合、評価値を得るためにはつぎのコマンドを使用しま す。

value = eval(LitVal)

一般に、ほとんどのパラメータは Evaluate の Assignment タイプを使用します。

最初のプロンプトの作成

リストに最初のプロンプトを作成するには、Prompt フィールドにプロンプトを 入力し、Variable フィールドにパラメータ値を含むための変数を入力し、コント ロールのスタイルと割り当てタイプを選択します。

プロンプトの挿入

リストにプロンプトを挿入するには、つぎのようにします。

- 1 挿入したい新しいプロンプトのすぐ **ア**に表示するプロンプトを選択し、プロ ンプトリストの左側の Add ボタンをクリックします。
- プロンプトに対するテキストを Prompt フィールドに入力します。パラメータ 値をもつための変数を Variable フィールドに入力します。

プロンプトの編集

既存のプロンプトを編集するには、つぎのようにします。

- 1 リスト内のプロンプトを選択します。プロンプト、変数名、コントロールの スタイル、割り当てタイプはリストの下のフィールドに表示されます。
- 適切な値を編集します。フィールドの外部をマウスでクリックするか、Enter または Return キーを押すと、Simulink はプロンプトを更新します。

プロンプトの削除

リストからプロンプトを削除するには、つぎのようにします。

- 1 削除したいプロンプトを選択します。
- 2 プロンプトリストの左側の Delete ボタンをクリックします。

プロンプトの移動

リスト内でプロンプトを移動するには、つぎのようにします。

- 1 移動したいプロンプトを選択します。
- 2 プロンプトリスト内でプロンプトを1つ上に移動するには、プロンプトリストの左側のUpボタンをクリックします。プロンプトを下に移動するには、 Downボタンをクリックします。

コントロールのタイプ

Simulink では、パラメータ値を入力したり選択したりする方法を選択することが できます。エディットフィールド、チェックボックス、ポップアップコントロー ルの3つのコントロールのスタイルを作成することができます。たとえば、つぎ の図は3つのコントロールのスタイルすべてを使用した、マスクダイアログボッ クスのパラメータ領域を示しています(ポップアップは開いています)。



エディットコントロールの定義

*エディットフィールド*では、パラメータ値をフィールドに入力することができます。つぎの図は、サンプルのエディットコントロールに対するプロンプトをどの ように定義したかを示しています。

Prompt:	Frequency:	Control type:	Edit 🗾
Variable:	freq	Assignment:	Evaluate 🔹

パラメータ(freq)に関連する変数の値は、プロンプトに対して定義されている Assignment タイプによって決まります。

Assignment	值
Evaluate	フィールドに入力される式の評価結果
Literal	フィールドに入力される実際の文字列

チェックボックスコントロールの定義

*チェックボックス*では、チェックボックスを選択するか選択解除するかによって2つの選択肢のいずれかを選ぶことができます。つぎの図は、サンプルのチェックボックスコントロールをどのように定義したかを示しています。

Prompt:	Show label	Control type:	Checkbox 💽
Variable:	label	Assignment:	Evaluate 💌

パラメータ (label) に関連する変数の値は、チェックボックスが選択されている かどうかにより、またプロンプトに対して定義されている Assignment タイプに よって異なります。

チェックボッ クス	評価値	文字の値
チェックあ り	1	'on'
チェックな し	0	'off'

ポップアップコントロールの定義

ポップアップでは、取り得る値のリストからパラメータ値を選択することができます。Popup strings フィールドにリストを指定し、項目は垂直線 ()) で区切ります。つぎの図は、サンプルのポップアップコントロールをどのように定義したかを示しています。

Prompt:	Color:		Control type:	Popup 🗾
Variable:	color		Assignment:	Evaluate 💽
Popup strir	ngs:	rediblueigreeniyellow		

パラメータ (color) と関連する変数の値は、ポップアップリストから選択した項目と、プロンプトに対して定義されている Assignment タイプによって異なります。

Assignment	值
Evaluate	リストから選択した値のインデックス。1 から始まります。 たとえば、3 番目の項目を選択するとパラメータ値は 3 にな ります。
Literal	選択した値の文字列。3 番目の項目を選択するとパラメータ 値は 'green' になります。

マスクされたブロックパラメータに対するデフォルト値

マスクされたライブラリブロックのデフォルトのパラメータ値を変更するには、 つぎの手順に従います。

- 1 ライブラリをアンロックします。
- 2 ブロックを開いてそのダイアログボックスにアクセスし、希望するデフォル ト値を入力し、ダイアログボックスを閉じます。
- 3 ブロックライブラリを保存します。

ブロックをモデルにコピーして開くと、デフォルト値がブロックのダイアログ ボックスに表示されます。

ライブラリに関する詳細は、4-77ページの"ライブラリ"を参照してください。

変更可能なパラメータ

変更可能なパラメータは、実行時にユーザが変更できるマスクパラメータです。 マスクを作成するとき、すべてのパラメータは変更可能です。

MaskTunableValues パラメータでマスクパラメータのすべての値の後からの変更 を無効にしたり、再変更可能にしたりすることができます。このパラメータの値 は文字のセル配列で、各々マスクされたブロックパラメータの1つに関連しま す。最初のセルは最初のパラメータに関連し、2番目のセルは2番目のパラメー タに関連し、その他も同様の関連があります。パラメータが変更可能である場 合、関連するセルの値は on であり、そうでなければ off です。パラメータの再変 更を有効にしたり無効にしたりするためには、まず、get param を使ってセル配 列を得る必要があります。それから関連するセルを on や off に設定し、 set_param を使って MaskTunableValues パラメータを再設定します。たとえば、 つぎのコマンドは、現在選択しているマスクブロックの最初のパラメータの再変 更を無効にします。

ca = get_param(gcb, 'MaskTunableValues');

ca(1) = 'off'

set_param(gcb, 'MaskTunableValues', ca)

ブロックの変更可能なパラメータを変更した後、ブロックを保存することで変更 を保存します。

初期化コマンド

初期化コマンドでは、マスクワークスペースに常駐する変数を定義します。これ らの変数は、マスクに対して定義されているすべての初期化コマンド、マスクさ れたサブシステム内のブロック、およびブロックアイコンを描くコマンド(描画 コマンド)で使用することができます。

Simulink は、つぎの場合に初期化コマンドを実行します。

- モデルをロードする場合
- シミュレーションの開始またはブロック線図の更新を行う場合
- マスクブロックを回転した場合
- ブロックのアイコンを再描画する必要があり、そのプロットコマンドが初期 化コマンドで定義した変数に依存する場合

初期化コマンドは、有効な MATLAB 表現であり、MATLAB 関数、演算子、およびマスクワークスペースで定義した変数から構成されています。初期化コマンドは、基本ワークスペース変数にはアクセスできません。初期化コマンドはセミコロンで終了し、結果がコマンドウィンドウに表示されないようにします。

マスクワークスペース

Simulink は、つぎの場合、マスクワークスペース と呼ばれるローカルワークスペースを作成します。

- マスクに初期化コマンドが含まれている場合
- マスクがプロンプトを定義し、変数をそれらのプロンプトと関連付けている 場合

マスクワークスペースの内容には、マスクのパラメータに関連する変数と、初期 化コマンドで定義された変数が含まれます。

本章の前半で説明した mx + b の例では、Mask Editor は、ある変数をマスクパラ メータと関連付けることによって、マスクワークスペースに明示的に m と b を 作成します。下の図は、マスクダイアログボックスに入力した値をマスクワーク スペース内の変数に反映させ(実線)、基になるブロックがそれらの変数へアク セスする(破線)様子を示しています。



マスクワークスペースは、M-ファイル関数で使用するローカルワークスペース に似ています。基になるブロックのダイアログボックスに入力した式と Mask Editor で入力した初期化コマンドを、M-ファイル関数の各行とみなすことがで きます。そうすると、この " 関数 " に対するローカルワークスペースはマスク ワークスペースに対応します。

マスクされたサブシステムはワークスペースに階層を作成します。マスクされた ブロックのワークスペースは、モデルワークスペースのサブスペースとマスクさ れたブロックをもつ任意のブロックのワークスペースのサブスペースです。マス クされたブロックは、そのワークスペース階層に固有に定義されたすべての変数 にアクセスすることができます。同様に、マスクされたサブシステムのブロック も、マスクされたサブシステムのワークスペース階層に定義された任意の変数に アクセスすることができます。 変数がその階層より上の階層に定義されている場合、マスクされたブロックは最 もローカルな定義にのみアクセスすることができます。たとえば、モデル M が マスクされたサブシステム A を含んでおり、A はマスクされたサブシステム B を含んでいると仮定します。さらに、B は、A と M のワークスペースに存在す る変数 x を参照すると仮定します。この場合、x は A のワークスペースの値を参 照します。

注意 マスクされたブロックの初期化コードは、マスクされたブロックのローカ ルワークスペースに定義された変数にのみアクセスすることができます。

初期化コマンドのデバッグ

初期化コマンドは、以下の方法でデバッグすることができます。

- 初期化コマンドをセミコロンで終了しないで、その結果をコマンドウィンド ウにエコーさせます。
- 実行を停止して、キーボードに制御を与えるために、初期化コマンドに keyboard コマンドを追加します。詳細については、keyboard コマンドに対する ヘルプテキストを参照してください。
- MATLAB ウィンドウに、つぎのコマンドのいずれかを入力します。

dbstop if error dbstop if warning

初期化コマンドでエラーが発生すると、実行は停止され、マスクワークス ペースを調べることができます。詳細については、dbstop コマンドに対するへ ルプテキストを参照してください。

Icon ページ

Icon ページでは、マスクされたブロックのアイコンをカスタマイズすることが できます。Drawing commands フィールドにコマンドを指定することによって、 アイコンをカスタマイズすることができます。説明のテキスト、状態方程式、イ メージ、グラフィックスを示すアイコンを作成することができます。つぎの図 は、Icon ページを示しています。



描画コマンドは、マスクワークスペース内のすべての変数にアクセスできます。

描画コマンドは、テキストや1つまたは複数のプロットを表示したり、伝達関数 を示したりすることができます。複数のコマンドを入力すると、コマンドの結果 はコマンドの表示順序に従ってアイコン上に描かれます。

ブロックアイコン上でのテキストの表示

アイコン上にテキストを表示するには、つぎの描画コマンドのいずれかを入力します。

disp('text') or disp(variablename)

text(x, y, 'text')
text(x, y, stringvariablename)
text(x, y, text, 'horizontalAlignment', halign, 'verticalAlignment', valign)

fprintf('text') or fprintf('format', variablename)

port_label(port_type, port_number, label)

disp コマンドは、text または variablename の内容を、アイコン上にセンタリングして表示します。

text コマンドは、点 (x,y) で指定された位置に文字列 (text または stringvariablename の内容)を配置します。単位は Drawing coordinates パラメータ に依存します。詳細については、7-22 ページの"アイコンプロパティの制御"を参 照してください。

text コマンドではオプションとして、点 (x,y) に対応するテキストの水平方向と 垂直方向の両方もしくは、どちらか一方の配置を指定することができます。たと えばコマンド

text(0.5, 0.5, 'foobar', 'horizontalAlignment', 'center')

は、アイコン上の中心に "foobar" を配置します。

text コマンドは、つぎの水平方向配置オプションをもっています。

オプション	配置
left	指定した点にテキストの左端がきます
right	指定した点にテキストの右端がきます
center	指定した点にテキストの中心がきます

text コマンドは、つぎの垂直方向配置オプションをもっています。

オプション	配置
base	指定した点にテキストの基準線がきます。
bottom	指定した点にテキストの下部がきます。
middle	指定した点にテキストの中心線がきます。

オプション	配置
cap	指定した点にテキストの大文字線がきます。
top	指定した点にテキストの上部がきます。

fprintf コマンドは、アイコン上でセンタリングされた書式付きテキストを表示し、variablenameの内容と共に text を表示することができます。

注意 これらのコマンドは、対応する MATLAB 関数と同じ名前ですが、上に示 す機能のみを提供します。

複数のテキスト行を表示するには、行の区切りを示す \n を使用します。たとえば、下の図は、disp コマンドの2つのサンプルを示しています。

disp('Sample Mask') disp('Sample\nMask')



port_label コマンドを使って、アイコン上に端子ラベルの表示を指定します。記 法はつぎのようになります。

port_label(port_type, port_number, label)

ここで、port_type は 'input' か 'output'、port_number は整数、label は端子ラベルを 指定する文字列です。たとえば、コマンド

port_label('input', 1, 'a')

は、入力端子1のラベルとしてaを指定しています。

ブロックアイコン上でのグラフィックスの表示

1 つまたは複数の plot コマンドを入力することによって、マスクされたブロック アイコン上にプロットを表示することができます。plot コマンドのつぎのような 形式を使用することができます。

plot(Y); plot(X1,Y1,X2,Y2,...); plot(Y) は、ベクトル Y に対して、そのインデックスに対する各要素をプロットします。 Y が行列の場合、それは行列の各列をベクトルとしてプロットします。

plot(X1,Y1,X2,Y2,...)は、X1 に対してベクトル Y1、X2 に対してベクトル Y2 という ようにプロットします。ベクトルの組合わせは長さが同じでなければならず、リ ストは偶数のベクトルから構成されていなければなりません

たとえば、つぎのコマンドは、Sources ライブラリの Ramp ブロックに対するア イコン上に表示されるプロットを生成します。コマンドの下にアイコンを示しま す。

plot([0 1 5], [0 0 4])

Ramo

プロットコマンドは、NaN と inf 値を含むことができます。NaN や inf があると Simulink は描画をやめ、NaN や inf でないつぎの数値があると再描画します。

アイコン上のプロットの外観は、Drawing coordinates パラメータの値によって 異なります。詳細については、7-22ページの"アイコンプロパティの制御"を参照 してください。

Simulink は、ブロックアイコンに3つの疑問符(???)を表示し、つぎの状況で ワーニングを発します。

- ・ 描画コマンドで使用するパラメータに対する値がまだ定義されていない場合(
 たとえば、マスクを初めて作成し、値をマスクダイアログボックスにまだ入
 カしていない場合)。
- マスクされたブロックパラメータまたは描画コマンドの入力が正しくない場合。

マスク上でのイメージの表示

マスクダイアログ関数 image と patch を使って、マスクブロックアイコン上に Bitmap イメージを表示したり、パッチを描画したりできます。

image(a) はイメージ a を表示します。ここで、a は m × n × 3 の RGB 値の配列です。 Bitmap ファイルを読み込んだり、必要な行列形式に変換するために MATLAB コ マンド imread や ind2rgb を利用することができます。たとえば、

image(imread('icon.tif'))

MATLAB パス上の icon.tif と名付けれた TIF ファイルからアイコンイメージを読み込みます。

image(a, [x, y, w, h]) は、マスクの左下隅からの相対的な指定位置にイメージを作成 します。

image(a, [x, y, w, h], rotation) は、アイコンが回転したときにイメージを回転させる ('on') かそのまま回転させない ('off') かを指定します。デフォルトは 'off' です。

patch(x, y)は、座標ベクトル x, y で指定した形のソリッドパッチを作成します。 パッチの色は、現在の foreground カラーになります。

patch(x, y, [r g b]) は、ベクトル [r g b] で指定した色のソリッドパッチを作成します。 ここで、r は赤成分、g は緑成分、b は青成分です。たとえば

patch([0.51], [010], [100])

は、マスクアイコン上に赤の三角形を作成します。

ブロックアイコン上での伝達関数の表示

ブロックアイコン上に伝達関数方程式を表示するには、Drawing commands フィールドにつぎのコマンドを入力します。

dpoly(num, den) dpoly(num, den, 'character')

num と den は伝達関数の分子係数と分母係数のベクトルで、一般に初期化コマンドを用いて定義します。方程式は指定した character で表現します。デフォルトは s です。アイコンを描画すると、初期化コマンドが実行され、結果として得られる方程式がアイコン上に描かれます。

 連続系伝達関数を s のべき乗の降順で表示するには、つぎのように入力します。 dpoly(num, den)

たとえば、num = [0 0 1]; および den = [1 2 1]; の場合、アイコンはつぎのように表示されます。

1	
s ² +2s+1	

離散系伝達関数をzのべき乗の降順で表示するには、つぎのように入力します。

dpoly(num, den, 'z')

たとえば num = [0 0 1] および den = [1 2 1] の場合、アイコンはつぎのように表示

1	
z ² +2z+1	

されます。

離散形伝達関数を 1/2 のべき乗の昇順で表示するには、次のように入力します。
 dpoly(num, den, 'z-')

たとえば、上で定義した num および den の場合、アイコンはつぎのように表

z^{.2} 1+2z-1+z-2

示されます。

 零点 - 極 - ゲイン型伝達関数を表示するには、つぎのコマンドを入力します。 droots(z, p, k)

たとえば、つぎの値に対して、上記のコマンドは、下のようなアイコンを作 成します。

z = []; p = [-1 -1]; k = 1;

1 (s+1)(s+1)

4 番目の引数 ('z' または 'z-') を追加して、 z または 1/z で方程式を表現すること ができます。

パラメータが定義されていなかったり、アイコンを作成するときに値をもってい ない場合、Simulink は、アイコンに3つの疑問符(???)を表示します。パラメー タ値をマスクダイアログボックスに入力すると、Simulink は伝達関数を評価し、 結果として得られる方程式をアイコンに表示します。

アイコンプロパティの制御

マスクされたブロックのアイコンのプロパティは、Drawing commands フィール ドの下の選択肢によって制御することができます。 Icon frame ($\mathcal{P} \mathcal{T} \mathcal{T} \mathcal{D} \mathcal{D} \mathcal{D} \mathcal{D} \mathcal{D}$)

アイコンフレームは、ブロックを囲む長方形のことです。Icon frame パラメータ を Visible または Invisible に設定することによって、フレームの表示または非表 示を選択することができます。デフォルトでは、アイコンフレームを表示しま す。たとえば、つぎの図は AND ゲートブロックに対するフレームの表示 / 非表 示を示しています。



Icon transparency(アイコンの透明性)

アイコンは、アイコンの下にあるものを非表示にするか表示するかを、Opaque または Transparent によって設定することができます。デフォルトは Opaque で、端子ラベルなどの Simulink が描く情報を覆い隠します。つぎの図は、AND ゲートブロックに対する不透明 (opaque) アイコンと透明 (transparent) アイコンを 示したものです。透明アイコン上のテキストに注意してください。



Icon rotation(アイコンの回転)

ブロックを回転したり反転したりする場合、アイコンを回転するか反転するか、 あるいはそのオリジナルの方向に固定させたままにするかを選択することができ ます。デフォルトではアイコンを回転しません。アイコンの回転はブロックの端 子の回転と一致します。つぎの図は、ANDゲートブロックを回転するときに、 アイコンの回転について Fixed(固定)と Rotates(回転)を選択した結果を示し ています。



Y

Rotates

Drawing coodinates(描画座標)

このパラメータは、描画コマンドが使用する座標系を制御します。このパラメー タは、plot および text 描画コマンドのみに適用されます。選択肢 Autoscale, Normalized, Pixel の中から選ぶことができます。



 Autoscale は、アイコンの大きさをブロックフレーム内で自動的に調整します。 ブロックの大きさを変更すると、アイコンの大きさも変更されます。たとえ ば、下の図は、つぎのベクトルを用いて描画したアイコンを示しています。 X = [0 2 3 4 9]; Y = [4 6 3 5 8];



ブロックフレームの左下隅は (0,3) で、右上隅は (9,8) です。x 軸の範囲は 9 (0 から 9) で、y 軸の範囲は 5 (3 から 8) です。

Normalized は、左下隅が(0,0)で右上隅が(1,1)であるブロックフレーム内部にアイコンを描画します。0と1の間のXとYの値のみが表示されます。ブロックの大きさを変更すると、アイコンの大きさも変更されます。たとえば、下の図は、つぎのベクトルを用いて描画したアイコンを示しています。

X = [.0.2.3.4.9]; Y = [.4.6.3.5.8];



 Pixel は、ピクセルで表現した X と Y の値を用いてアイコンを描きます。ブロックの大きさを変更しても、アイコンの大きさは自動的には変更されません。 アイコンの大きさをブロックとともに強制的に変更するには、描画コマンドをブロックサイズで定義します。

つぎの例は、本章の前半で説明した mx + b のマスクされたサンプルサブシス テムに対して、アイコンを改良する方法を示しています。これらの初期化コ マンドは、ブロックの形状に関係なく、描画コマンドが正確なアイコンを生 成するためのデータを定義します。 pos = get_param(gcb, 'Position'); width = pos(3) - pos(1); height = pos(4) - pos(2); x = [0, width]; if (m >= 0), y = [0, (m*width)]; end if (m < 0), y = [height, (height + (m*width))]; end</pre>

このアイコンを生成する描画コマンドは、plot(x,y)です。

Documentation ページ

Documentation ページでは、マスクされたブロックに対するタイプや説明、ヘル プテキストを定義したり変更したりすることができます。つぎの図は、 Documentation ページのフィールドが、mx+bのサンプルのマスクブロックのダ イアログボックスにどのように対応しているかを示しています。

🛃 Mask Editor: untitled/mx + b 📃 🖂 🔀	
Icon Initialization Documentation	
Mask type: SampleMaskedBlock	Block Parameters: mx + b SampleMaskedBlock (mask)
Block description:	Models the equation for a line, y = mx + b. The slope and intercept are mask block parameters.
Models the equation for a line, y = mx + b. The slope and intercept are mask block parameters.	Parameters Slope: 3 Intercept: 2
Block help:	OK Cancel Help
Enter the slope (m) and intercept (b) in the block dialog parameter fields. The block generates y for a given input x.	
OK Cancel Unmask Help Apply	

Mask Type フィールド

マスクタイプは、ドキュメント化のためだけに用いるブロックの種類です。マス クタイプは、ブロックのダイアログボックスとブロックに対するすべての Mask Editor ページに表示されます。マスクタイプには任意の名前を選択することがで きます。Simulink がブロックのダイアログボックスを作成する場合、マスクタイ プの後に "(mask)" を追加して、マスクされたブロックと組み込みブロックを区別 します。

Block Description フィールド

ブロック説明は、マスクタイプの下のフレーム内の、ブロックのダイアログボッ クスに表示される情報テキストです。他の人が使用するシステムを設計している 場合、これはブロックの目的または機能を記述する最適の場所です。 Simulink は、長いテキスト行に対して自動的に改行を行います。Enter または Return キーを用いて強制的に改行することもできます。

Mask Help Text フィールド

マスクされたブロックのダイアログボックス上で Help ボタンが押されたときに 表示されるヘルプテキストを設定することができます。他の人が利用するモデル を作成する場合、ブロックの機能やパラメータの入力方法を説明するのに適した フィールドです。

マスクされたブロックのヘルプとしてユーザが作成したドキュメンテーションを 含めることができます。マスクされたブロックのヘルプテキストとして、つぎの ものを指定することができます。

- ・ URL 指定 (http:, www, file:, ftp:, mailto: で始まる文字列)
- web コマンド(ブラウザを起動)
- ・ eval コマンド (MATLAB 文字列を評価)
- Web ブラウザで表示される Static テキスト

Simulink は、マスクされたブロックのヘルプテキストの1行目を調べます。URL の指定、web コマンド、eval コマンドを検出した場合は、指定された通りにブ ロックのヘルプにアクセスします。そうでない場合は、マスクされたブロックの ヘルプテキストの完全な内容がプラウザに表示されます。

つぎの例は、可能なコマンドを示しています。

web([docroot '/My Blockset Doc/' get_param(gcb,'MaskType')... '.html']) eval('!Word My_Spec.doc') http://www.mathworks.com file:///c:/mydir/helpdoc.html www.mathworks.com

Simulink は、長いテキスト行を自動的に改行します。

Self-Modifying マスクブロックの作成

マスクブロックは、ユーザ入力に応じてブロック自身を変更することができま す。特に、マスクされたブロックは基となるシステムブロックの内容を変更する ことができ、ユーザ入力を基にそれらのブロックのパラメータを設定することが できます。たとえば、ユーザ設定に応じて、入力や出力を追加または削除するブ ロックを作成することができます。

self-modifying マスクブロックの作成時には、MaskSelfModifiable パラメータを 'on'に設定しなければなりません。そうでない場合は、Simulink はブロックがブ ロック自身を変更しようとしたとき、つまりマスクブロックのワークスペースの コードが基になるシステムブロックからブロックを追加あるいは削除しようとし たとき、または基となるシステムブロック内の任意のブロックのパラメータを変 更しようとしたときに、エラーを発生します。

MaskSelfModifiable パラメータを設定するには、self-modifying ブロックを選択し、MATLAB プロンプトでつぎのコマンドを入力します。

set_param(gcb, 'MaskSelfModifiable', 'on');

それから、ブロックを保存します。

マスクブロックに対する動的ダイアログの作成

Simulink では、ユーザ入力に対応して変化するマスクブロックのためのダイアロ グを作成することができます。マスクダイアログの内容は、この方法でつぎに示 すような特徴を変更することができます。

・ パラメータコントロールの可視、不可視

パラメータを変更することで、他のパラメータのためのコントロールを表示 したり、非表示にしたりできます。コントロールが現れたり消えたりすると、 ダイアログは拡張されたり縮小されたりします。

・ パラメータコントロールの有効状態

パラメータを変更することで、他のパラメータのためのコントロールを入力 可能にしたり、入力不可能にしたりできます。Simulink はコントロールが無効 であることを視覚的に示すために無効なコントロールを灰色にします。

パラメータ値

パラメータを変更することで、対応するパラメータを適切な値に設定するこ とができます。

動的マスクダイアログを作成するためには、Simulinkのset_param コマンドと組 み合わせてマスクエディタを利用します。厳密には、まずマスクエディタを使っ て静的なものと動的なものの両方のすべてのダイアログパラメータを定義しま す。つぎに MATLAB コマンドラインで Simulinkのset_param コマンドを使って、 ユーザ入力に対するダイアログの応答を定義するコールバックファンクションを 定義します。最後に、動的マスクダイアログを確定するためにマスクサプシステ ムを含むモデルやライブラリを保存します。

マスクブロックダイアログのパラメータの設定

Simulink は、マスクブロックのダイアログの現在の状態を定義するマスクブロッ クパラメータの集合を定義します。これらのパラメータのいくつかを確認したり 設定するためにマスクエディタを利用することができます。Simulinkの get_param コマンドと set_param コマンドでもマスクダイアログパラメータを確認 し、設定します。この利点は何でしょうか? set_param コマンドを使ってパラ メータを設定することができ、従って、ダイアログがオープンする時のダイアロ グの表示を変更することができます。これにより、動的マスクダイアログの作成 が可能です。

たとえば、MATLAB コマンドラインで set_param コマンドを実行することにより、ユーザ定義のパラメータの値を変更した時に呼び出されるコールバック関数

を定義することができます。マスクダイアログパラメータの既存の設定値を変更 するために、set_param コマンドでコールバック関数を利用することができます。 従って、たとえば、ユーザ定義のパラメータコントロールを不可視にしたり、可 視にしたり、有効にしたり、無効にしたりできます。

定義済みマスクダイアログパラメータ

Simulink は、つぎの定義済みパラメータをマスクダイアログと関連付けます。

MaskCallbacks

このパラメータの値は、ダイアログのユーザ定義パラメータのコントロールに対 するコールバック表現を指定する文字列のセル配列です。1番目のセルは1番目 のパラメータコントロールに対するコールバックを定義し、2番目のセルは2番 目のパラメータコントロールに対するコールバックを定義し、その他も同様に定 義します。コールバックは有効な MATLAB の表現法である必要があり、Mファ イルのコマンドを含む表現法でも可能です。このことは、複雑なコールバックを Mファイルとして組み込むことができるということを意味します。

マスクダイアログにコールバックを設定する最も容易な方法は、モデルウィンド ウやライブラリウィンドウで関連するマスクダイアログを選択し、その後 MATLAB コマンドラインで set_param コマンドを実行することです。たとえば、 つぎのコード

set_param(gcb,'MaskCallbacks',{'parm1_callback', ",... 'parm3_callback'});

は、現在選択されているブロックに対するマスクダイアログの1番目と3番目の パラメータに対してコールバックを定義します。コールバックの設定を保存する ためには、マスクブロックを含むモデルやライブラリを保存します。

MaskDescription

このパラメータの値は、このブロックの記述を定義する文字列です。このブロックの設定により、マスクブロックの記述を動的に変更できます。

MaskEnables

このパラメータの値は、このダイアログに対するユーザ定義パラメータコント ロールの有効状態を定義するための文字列のセル配列です。1番目のセルは1番 目のパラメータに対するコントロールの有効状態を、2番目のセルは2番目のパ ラメータに対するコントロールの有効状態を、その他もそれぞれのコントロール の有効状態を定義します。設定値 'on' は関連するコントロールがユーザ入力可能 であることを示し、'off' はユーザ入力不可能であることを示します。 コールバックでこのパラメータを設定することで、動的にユーザ入力を可能にしたり不可能にしたりできます。たとえば、コールバックで、つぎのコマンド

set_param(gcb,'MaskEnables',{'on','on','off'});

を実行すると、現在オープンしているブロックダイアログの3番目のコントロー ルが無効になります。Simulink は、コントロールが無効であることを視覚的に示 すためにコントロールを灰色にします。

MaskPrompts

このパラメータの値は、ユーザ定義パラメータに対するプロンプトを定義する文 字列のセル配列です。1番目のセルは1番目のパラメータに対するプロンプト を、2番目のセルは2番目のパラメータに対するプロンプトを、その他もそれぞ れのプロンプトを定義します。

MaskType

このパラメータの値は、このダイアログに関連するブロックのマスクタイプで す。

MaskValues

このパラメータの値は、このダイアログに対するユーザ定義パラメータの値を定 義する文字列のセル配列です。1番目のセルは1番目のパラメータに対する値 を、2番目のセルは2番目のパラメータに対する値を、その他もそれぞれの値を 定義します。

MaskVisibilities

このパラメータの値は、このダイアログに対するユーザ定義パラメータコント ロールの可視性を定義する文字列の配列です。1番目のセルは1番目のパラメー タに対するコントロールの可視性を、2番目のセルは2番目のパラメータに対す るコントロールの可視性を、その他もそれぞれのコントロールの可視性を定義し ます。設定値 'on' は関連するコントロールが可視であることを示し、'off' は不可 視であることを示します。

コントロールに対するコールバックでこのパラメータを設定することにより、動 的にユーザ定義パラメータコントロールを隠したり見せたりすることができま す。たとえば、コールバックでつぎのコマンド

set_param(gcb,'MaskVisibilities',{'on','off','on'});

を実行すると、現在選択されているブロックの2番目のユーザ定義マスクパラ メータに対するコントロールを隠します。Simulinkはコントロールを見せたり隠 したりするのに対応して、ダイアログを拡張したり縮小したりします。

条件付きで実行されるサブ システム

はじめに	•		•			•		•	. 8-2
Enabled サブシステム									. 8-3
Enabled サブシステムの作成									. 8-3
Enabled サブシステムに組み込み可能なブロッ	ク	•	•		•	•	•	•	. 8-5
Triggered サプシステム									. 8-8
Triggered サブシステムの作成									. 8-9
Function-Call サブシステム									8-10
Triggered サブシステムに組み込み可能な ブロ	ッ	ク	•	•		•	•	•	8-10
Triggered and Enabled サプシステム									.8-12
Triggered and Enabled サブシステムの作成									8-12
Triggered and Enabled サブシステムのサンプル									8-13
交互に実行するサブシステムの作成									8-13

はじめに

*条件付きで実行されるサブシステムとは、*実行が入力信号の値に依存するよう なサプシステムです。サプシステムを実行するかどうかを制御する信号は、*制御 信号と*呼ばれます。この信号は、*制御入力端子*から Subsystem ブロックに入りま す。

条件付きで実行されるサブシステムは、実行が他の成分に依存する成分を含む複 雑なモデルを構築するような場合、非常に便利です。

Simulink は、3種類の条件付きで実行されるサブシステムをサポートします。

- enabled サブシステムは、制御信号が正のときに実行されます。このサブシステムは、制御信号が(負から正の方向に)ゼロを横切る時間ステップで実行を開始し、制御信号が正の状態である間実行を継続します。Enabled サブシステムの詳細については、8-3 ページの "Enabled サブシステム"で説明します。
- triggered subsystem サブシステムは、"トリガイベント"が発生するたびに実行されます。トリガイベントは、連続系または離散系のどちらでも、トリガ信号が正から負に変化する瞬間、負から正に変化する瞬間、またはその両方の瞬間に生じます。Triggered サブシステムの詳細については、8-8ページの "Triggered サブシステム"で説明します。
- triggered and enabled サブシステムは、トリガイベントが発生したときの時間ステップで、Enable 制御信号が正の値の場合に実行されます。Triggered and Enabled サブシステムの詳細については、8-12 ページの"Triggered and Enabled サブシステム"で説明します。
Enabled サブシステム

Enabled サブシステムは、制御信号が正の値である各シミュレーションステップ で実行されるサブシステムです。

Enabled サブシステムは、スカラまたはベクトルの値をとることができる、単一の制御入力をもっています。

- 入力がスカラの場合、入力値がゼロより大きい場合にサブシステムは実行されます。
- 入力がベクトルの場合、ベクトル要素の*いずれか*がゼロより大きい場合にサ ブシステムは実行されます。

たとえば、制御入力信号が正弦波の場合、サブシステムはつぎの図に示すように 実行可能と実行不能の状態を交互に繰り返します。上向きの矢印は実行可能、下 向きの矢印は実行不能を表します。



Simulink は、ゼロクロッシング勾配法を用いて、実行可能の状態が発生するかど うかを決定します。信号がゼロを横切る瞬間の勾配が正の場合、サブシステムは 実行可能状態です。勾配がゼロクロッシングで負の場合、サブシステムは実行不 能状態です。

Enabled サブシステムの作成

Enabled サブシステムは、Signals & Systems ライブラリから Enable ブロックをサ ブシステムにコピーすることによって作成します。Simulink は、Enable ブロック の記号と Enable 制御入力端子を Subsystem ブロックアイコンに追加します。.



サブシステムが実行不能なときの出力値の設定

Enabled サブシステムは、実行不能である間は実行されませんが、出力信号は依 然として他のブロックで利用できます。Enabled サブシステムが実行不能になっ ている間、サブシステム出力を前の値に保持するか、初期条件にリセットするか を選択することができます。

各 Outport ブロックのダイアログボックスをオープンし、下のダイアログボック スに示すように、Output when disabled パラメータに対する選択肢のいずれかを 選びます。

- held を選択すると、出力として最新の値が保持されます。
- reset を選択すると、出力は初期条件にリセットされます。出力の初期値は Initial output に設定します。

Block Parameters: Out1 🛛 🗶	
Outport	
Provide an output port for a subsystem or model. The 'Output when disabled' and 'Initial output' parameters only apply to conditionally executed subsystems. When a conditionally executed subsystem is disabled, the output is either held at its last value or set to the 'Initial output'. The 'Initial output' parameter can be specified as the empty matrix. [] in which case	
the initial output is equal to the output of the block feeding the outport.	
Parameters	
Port number:	サゴンファノゼウケス化スキス明の
	リノンステムが美行不能でのる間の Outpurt 出力を設定するためのオプション
Output when disabled: held	を選択します。
Initial output:	
	リーットはの辺如名供上は
·	リセット時の初期余件と値。
OK Cancel <u>H</u> elp <u>Apply</u>	

サブシステムが再び実行可能になるときの状態の設定

Enabled サブシステムが実行されるときに、サブシステムの状態を前の値に保持 するか、初期条件にリセットするかを選択することができます。

そのためには、Enable ブロックのダイアログボックスを開き、下のダイアログ ボックスに示すように、States when enabling パラメータの選択肢のいずれかを選 びます。

- held を選択すると、状態として最新の値が保持されます。
- resetを選択すると、状態は初期条件にリセットされます。

Block Parameters: Enable	
Enable Port	
Place this block in a subsystem to create an enabled subsystem.	
Parameters	
States when enabling: held	ナプションを溜切り、サブシュニノが声が安
Show output port held reset	- 行可能になるときの状態を設定します。
OK Cancel Help Apply	

Enable 制御信号の出力

Enable ブロックのダイアログボックスのオプションを用いると、Enable 制御信 号を出力することができます。制御信号を出力するには、Show output port チェックボックスを選択します。

Block Parameters: Enable	
Place this block in a subsystem to create an enabled subsystem.	
Parameters States when enabling: held Show output port	このチェックボックスを選択して出力端子
OK Cancel <u>H</u> elp <u>Apply</u>	

この機能によって、制御信号を Enabled サブシステムに渡すことができ、これ は、Enabled サブシステム内部のロジックが制御信号に含まれる値(1つまたは複 数)に依存するようなところで有効です。

Enabled サブシステムに組み込み可能なブロック

Enabled サブシステムは、連続系または離散系を問わず、任意のブロックを含む ことができます。Enabled サプシステム内の離散ブロックは、サプシステムが実 行されるときのみ、しかもそのサンプル時間がシミュレーションサンプル時間と 同期がとれているときのみ実行されます。Enabled サブシステムおよびモデル は、共通のクロックを使用します。

注意 Enabled サブシステムは、GoTo ブロックを含むことができます。しかし、 Enabled サブシステムでは状態端子のみが GoTo ブロックに接続できます。 Enabled サブシステムでの GoTo ブロックの使用法の例は、Simulink デモモデル、 clutch を参照してください。 たとえば、つぎのシステムには、4つの離散ブロックと1つの制御信号が含まれています。離散ブロックはつぎのとおりです。

- サンプル時間が 0.25 秒の Block A
- サンプル時間が 0.5 秒の Block B
- サンプル時間が 0.125 秒の、Enabaled サブシステム内の Block C
- ・ サンプル時間が 0.25 秒の、同じく Enabaled サブシステム内の Block D

Enable 制御信号は、Signal E というラベルの付けられた Pulse Generator ブロック によって生成され、0.375 秒で0から1に変化し、0.875 秒で0に戻ります。.



下の図は、離散ブロックが実行される時間を示しています。



Block A と Block B は、Enabled サブシステムの一部ではないので、Enable 信号と は独立して実行されます。Enable 信号が正になると、Block C と D は Enable 信 号が再びゼロになるまで、割り当てられたサンプリングレートで実行されます。 Enable 信号がゼロになる 0.875 秒ではブロック C が実行されないことに注意して ください。

Triggered サブシステム

Triggered サブシステムは、トリガイベントが発生するたびに実行されるサブシ ステムです。

Triggered サブシステムには、サブシステムを実行するかしないかを決定する *リガ入力*と呼ばれる1つの制御入力があります。3種類のトリガイベントから選 択して、Triggered サプシステムの実行を強制的に開始させることができます。

- rising は、制御信号が負またはゼロから正の値(または初期値が負の場合はゼロ)に変化するときにサブシステムを実行します。
- fallingは、制御信号が正あmたはゼロから負の値(または初期値が正の場合はゼロ)に変化するときにサブシステムを実行します。
- either は、信号が rising または falling のときにサブシステムを実行します。

たとえば、つぎの図は与えられた制御信号に対して、rising(R) および falling(F) のトリガイベントが発生する瞬間を示しています。



Triggered サブシステムの簡単な例を示します。



この例のサブシステムは、矩形波トリガ制御信号が0から正の値に変化するとき をトリガイベントとします。

Triggered サブシステムの作成

Triggered サブシステムは、Signals & Systems ライブラリから Trigger ブロックを サプシステムにコピーすることによって作成します。Simulink は、Trigger ブ ロックの記号とトリガ制御入力端子を Subsystem ブロックアイコンに追加しま す。



Subsystem

トリガタイプを選択するには、Trigger ブロックのダイアログボックスをオープンし、下のダイアログボックスに示すように Trigger type パラメータの選択肢のいずれかを選びます。

- rising を選択すると、トリガ信号が正の方向でゼロを横切る瞬間をトリガイベントとします。
- falling を選択すると、トリガ信号が負の方向でゼロを横切る瞬間をトリガイベントとします。
- either を選択すると、トリガ信号が正または負いずれかの方向でゼロを横切る 瞬間をトリガイベントとします。.

Block Parameters: Trigger ISI Trigger Port Place this block in a subsystem to create a triggered subsystem.	
Parameters Trigger type: rising □ Show outp rialing Image: Show outp Output: delter OK Cancel Help Apply	— 選択肢から

選択肢からトリガタイプを選びます。

Simulink は、Trigger ブロックと Subsystem ブロック上で、それぞれのトリガイベントを異なる記号を用いて示します。つぎの図は、Subsystem ブロック上のトリガ記号を示したものです。



トリガイベント間の出力と状態

Enabled サブシステムと異なり、Triggered サブシステムは、トリガイベント間で それらの出力を常に最後の値に保持します。また、Triggered サブシステムはトリ ガイベントが生じたときにそれらの状態をリセットすることはできません。任意 の離散プロックの状態は、トリガイベントの間で保持されます。

トリガ制御信号の出力

Trigger ブロックのダイアログボックス上のオプションでは、トリガ制御信号を 出力することができます。制御信号を出力するには、Show output port チェック ボックスを選択します。

Block Parameters: Trigger × Trigger Port Place this block in a subsustem to create a triggered subsustem	
Parameters Trigger type: ising Show output port Output data type: auto	- チェックボックスを選択して出力端子を表示し ます。
OK Cancel <u>H</u> elp <u>Apply</u>	

Output data type フィールドは、出力信号のデータタイプを auto, int8, double として定義することができます。auto オプションを指定すると、出力信号のデータタイプを信号が接続される端子のデータタイプ (int8 か double) に設定します。

Function-Call サブシステム

信号の値の代わりに S-function の内部のロジックで実行が決定される Triggered サブシステムを作成することができます。これらのサブシステムは、 *function-call* サブシステムと呼ばれます。関数呼び出しサブシステムに関する詳 細は、Writing S-Functions ガイドをご参照ください。

Triggered サブシステムに組み込み可能な ブロック

Triggered システムは、シミュレーション中の特定な場合でのみ実行されます。 その結果、Triggered サブシステムで使われるのに適したブロックは、以下のも ののみです。

 Logical Operator ブロックまたは Gain ブロックなどの、継承したサンプル時間を もつブロック サンプル時間が -1 に設定された離散ブロック。-1 はサンプル時間が接続されるブロックから継承されることを示しています。

Triggered and Enabled サブシステム

条件付きで実行される3番目の種類のサブシステムは、2つのタイプの条件付き 実行を組み合わせたものです。triggered and enabled サブシステムと呼ばれるこの タイプのサブシステムの挙動は、下の流れ図に示すように Enabled サブシステム と Triggered サブシステムを組み合わせたものです。.



Triggered and Enabled サブシステムは、Enable 入力端子と Trigger 入力端子の両方 を含んでいます。トリガイベントが発生すると、Simulink は Enable 入力端子を チェックして Enable 制御信号を評価します。その値がゼロより大きいと、 Simulink はサブシステムを実行します。両方の入力がベクトルの場合は、少なく とも各ベクトルの1つの要素が非ゼロであれば、サブシステムは実行されます。

サブシステムは、トリガイベントが発生する時間ステップごとに実行されます。

Triggered and Enabled サブシステムの作成

Triggered and Enabled サブシステムは、Signal & Systems ライブラリから Enable ブ ロックと Trigger ブロックの両方を既存のサブシステム内にドラッグすることに よって作成します。Simulink は、Enable ブロックの記号と Trigger ブロックの記 号、および Enable 制御入力端子と Trigger 制御入力端子を、Subsystem ブロック アイコンに追加します。



Enabled サブシステムに対する場合と同様、Triggered and Enabled サブシステムが 実行不能なときの出力値を設定することができます。詳細については、8-4 ページの"サブシステムが実行不能なときの出力値の設定"を参照してください。また、サブシステムが再び実行可能になるときの状態の値を指定することもできます。詳細については、8-4 ページの"サブシステムが再び実行可能になるときの状態の設定"を参照してください。

Enable ブロックと Trigger ブロックのパラメータは別々に設定してください。手順は個々のブロックについて記述されている手順と同じです。

Triggered and Enabled サブシステムのサンプル

Triggered and Enabled サブシステムの簡単な例を、つぎのモデルに示します。



交互に実行するサブシステムの作成

Merge ブロックで結合して条件付き実行サブシステムを使うことにより、モデルの現在の状態に依存して交互に実行するサブシステムの集合を作成することができます。たとえば、つぎの図は2つの Enabled サブシステムと Merge ブロックを



用いたインバータのモデルを示しています。つまり、AC電流をパルス DC電流 に変換する装置をモデル化しています。

この例では、"pos" とラベル付けされたブロックが AC 正弦波が正のレベルの時 に実行され、正弦波をそのまま出力します。"neg" とラベル付けされたブロック は正弦波が負のレベルの時に実行され、正弦波を反転して出力します。Merge ブ ロックは、現在実行されているブロックの出力を Mux ブロックに受け渡し、 Mux ブロックでは、オリジナルの正弦波と重ねあわせて Scope ブロックに出力 し、最終的につぎの表示を作成します。



ブロックリファレンス

各プロックのリファレンスページの内容				•	•	•		9-2
Simulink プロックライブラリ								9-3

各ブロックのリファレンスページの内容

ブロックはアルファベット順に並べられ、つぎの情報を含んでいます。

- ブロック名、アイコン、およびブロックを含むブロックライブラリ
- ブロックの目的
- ブロックの使用法
- ブロックに入力可能またはブロックが生成するデータタイプと数値タイプ(複 素数または実数)
- ブロックのダイアログボックスとパラメータ
- ブロックの特性。この特性の中には以下に挙げられる特性のうち当該ブロックに適用できるもの全てまたは一部が含まれています。
 - 直接フィードスルー ブロックまたはブロックの端子のいずれかが、直接 フィードスルーになるか否かをチェック。詳細は、3-18 ページの"代数ルー プ"を参照してください。
 - サンプル時間 ブロックのサンプル時間の決定法、(離散ブロックや連続ブロックの場合と同様に)ブロック自身が設定するか、またはそのブロックに入力してくるものを継承するか、そのブロックが出力するものを継承するかによって決定されます。詳細は、3-23ページの"サンプル時間"を参照してください。
 - スカラ拡張 スカラ値が、ベクトルに拡張されるかどうかを示します。ブロックの中には、スカラ入力及び/またはパラメータを適切に拡張するものもあります。詳細は、4-36ページの"入力とパラメータのスカラ拡張"を参照してください。
 - 状態 離散状態と連続状態の数
 - ベクトル化 当該ブロックにベクトル信号を入力できるのか、そして / また はベクトル信号を生成するのかどうかについての情報がここで示されます。
 詳細については、4-30ページの"信号の利用"を参照してください。
 - ゼロクロッシング 当該ブロックが状態事象 (state event) を探知するかどうか を示します。詳細については、3-14 ページの"ゼロクロッシングの検出"を参 照してください。

Simulink ブロックライブラリ

Simulink は、個々のブロックの挙動に従って、ブロックライブラリに分けてブロックを提供しています。simulink ウインドウは、ブロックライブラリアイコンと名前を表示します。

- Sources ライブラリは、信号を生成するブロックで構成されます。
- Sinks ライブラリは、ブロック出力を表示または出力するブロックで構成されます。
- Discrete ライブラリは、離散時間要素を記述するブロックで構成されます。
- Continuous ライブラリは、線形関数を記述するブロックで構成されます。
- Math ライブラリは、一般的な数学関数を記述するブロックで構成されます。
- *Functions* & *Tables* ライブラリは、一般的な関数及びテーブルルックアップ演算を記述するブロックで構成されます。
- Nonlinear ライブラリは、非線形関数を記述するブロックで構成されます。
- Signal & Systems ライブラリは、マルチプレックスとデマルチプレックス、外部 入力 / 出力の実行、モデルの他のパーツへのデータ転送、サブシステムの生 成、他の関数の実行、といった操作を行うブロックで構成されます。
- Blocksets and Toolboxes ライブラリは、特殊ブロックの Extra ブロックライブラリ で構成されます。
- Demos ライブラリは、有用な MATLAB 及び Simulink デモで構成されます。

注意 Simulink ライブラリブラウザ (Windows のみ) を使用するか、MATLAB コ マンドで simulink3 と入力すると、ブロックライブラリを表示、ブラウズするこ とができます。

つぎの表は Blocksets and Toolboxes と Demos ライブラリを除く、すべてのライブ ラリのリストです。

表 9-1: Sources ライブラリブロック

プロック名	目的
Band-Limited White Noise	連続システムに白色ノイズを導入
Chirp Signal	周波数が増加する正弦波の発生

プロック名	目的
Clock	シミュレーション時間を表示、設定
Constant	定数値の発生
Digital Clock	指定サンプリング間隔でのシミュレーション 時間の発生
Digital Pulse Generator	一定間隔でパルスを生成
From File	ファイルからデータの読み込み
From Workspace	ワークスペースに定義した行列からのデータ の読み込み
Pulse Generator	一定間隔でのパルス波の発生
Ramp	一定の割合で増加または減少する信号の発生
Random Number	正規分布乱数の発生
Repeating Sequence	反復可能な任意の信号を発生
Signal Generator	種々の波形の発生
Sine Wave	正弦波の発生
Step	Step 関数の発生
Uniform Random Number	一様分布乱数の発生

表 9-1: Sources ライブラリブロック (Continued)

表 9-2: Sinks ライブラリブロック

Block Name	目的
Display	入力値の表示
Scope	シミュレーション中に生成される信号の表示
Stop Simulation	入力が非ゼロの場合に、シミュレーションを 停止

Block Name	目的
To File	ファイルへのデータの書き出し
To Workspace	データをワークスペース内の行列に書き出し
XY Graph	MATLAB の Figure ウィンドウを使って、信 号の X-Y プロット表示

表 9-2: Sinks ライブラリブロック (Continued)

表 9-3: Discrete ライブラリブロック

プロック名	目的
Discrete Filter	IIR フィルタと FIR フィルタの実現
Discrete State-Space	離散状態空間システムの実現
Discrete-Time Integrator	信号の離散時間積分の実行
Discrete Transfer Fcn	離散伝達関数の実現
Discrete Zero-Pole	極 - 零点型で指定された離散伝達関数の実現
First-Order Hold	1次サンプルアンドホールドの実現
Unit Delay	信号の1サンプル周期の遅れ
Zero-Order Hold	1 サンプル周期のゼロ次ホールドの実現

表 9-4: Continuous ライブラリプロック

ブロック名	目的
Derivative	入力の時間微係数の出力
Integrator	信号を積分
Memory	前回の積分ステップからのブロック入力を出 力

プロック名	目的
State-Space	線形状態空間システムを実現
Transfer Fcn	線形伝達関数を実現
Transport Delay	設定した時間だけ入力を遅延
Variable Transport Delay	可変時間で入力を遅延
Zero-Pole	極と零点の項で設定された伝達関数を実現

表 9-4: Continuous ライブラリブロック (Continued)

表 9-5: Math ライブラリプロック

プロック名	目的
Abs	入力の絶対値を出力
Algebraic Constraint	入力信号をゼロに設定
Bitwise Logical Operator	符号なし整数信号のビットを論理マスク、反 転、シフト
Combinatorial Logic	真理値表を実現
Complex to Magnitude-Angle	複素数入力信号の位相と大きさを出力
Complex to Real-Imag	複素数入力信号を構成する実部と虚部の出力
Derivative	入力の時間微係数を出力
Dot Product	内積を出力
Gain	ブロック入力の乗算
Logical Operator	入力に指定した論理演算を実行
Magnitude-Angle to Complex	大きさと位相角を複素数信号に変換
Math Function	数学関数を実行
Matrix Gain	入力と行列の乗算を計算

ブロック名	目的
MinMax	最小または最大入力値を出力
Product	ブロック入力の積または商を出力
Real-Imag to Complex	実部と虚部の入力から複素数信号を出力
Relational Operator	指定した比較演算を入力に適用
Rounding Function	丸め関数を適用
Sign	入力の符号を表示
Slider Gain	スライダを使って、スカラゲインを変化
Sum	入力の総和を出力
Trigonometric Function	三角関数を適用

表 9-5: Math ライブラリブロック (Continued)

表 9-6: Functions & Tables ライブラリブロック

ブロック名	目的
Direct Look-Up Table (n-D)	
Fcn	指定された式を入力に適用
Look-Up Table	入力の区分的線形写像を実行
Look-Up Table (2-D)	2 入力の区分的線形写像を実行
Look-Up Table (n-D)	複数入力の区分敵線形またはスプライン写像 を実行
MATLAB Fcn	MATLAB 関数または式を入力に適用
S-Function	S- ファンクションにアクセス

ブロック名	目的
Backlash	遊びのあるシステムの挙動のモデル化(バッ クラッシュ)
Coulomb & Viscous Friction	零点で不連続、その他では線形ゲインを伴う 型のモデル化
Dead Zone	ゼロ出力の領域を生成(デッドゾーン)
Manual Switch	2 つの入力の間での切り替え
Multiport Switch	ブロック入力の選択
Quantizer	指定された間隔で入力を離散化
Rate Limiter	信号の変化率を制限
Relay	2 つの定数間で出力の切り替え
Saturation	信号の範囲を制限
Switch	2 つの入力間で切り替え

表 9-7: Nonlinear ライブラリブロック

表 9-8: Signals & Systems ライブラリブロック

プロック名	目的
Bus Selector	入力バスから入ってくる信号の選択
Configurable Subsystem	指定したライブラリから選択した特定のブ ロックのみを表現
Data Store Memory	共有データストアを定義
Data Store Read	共有データストアからデータを読み込み
Data Store Write	共有データストアにデータを書き込み

プロック名	目的
Data Type Conversion	信号を他のデータタイプに変換
Demux	ベクトル信号を複数の出力信号に分解
Enable	Enable 端子をサブシステムに付加
From	Goto ブロックからの入力を受け取り
Goto	ブロック入力を From ブロックに渡す
Goto Tag Visibility	Goto ブロックタグの範囲を定義
Ground	未接続の入力端子を接地
Hit Crossing	クロッシングポイントの検出
IC	信号の初期値を設定
Inport	サブシステムまたは外部入力に対する入力端 子を生成
Matrix Concatenation	配列入力の結合
Merge	複数入力ラインを一本のスカラ出力ラインに 結合
Model Info	モデル内に作成に関する情報を表示
Mux	複数の入力ラインから 1 つのベクトルライン を作成
Outport	サブシステムまたは外部出力に対する出力端 子を生成
Reshape	信号の大きさを変更
Probe	入力信号の幅、サンプル時間、及び/または 信号のタイプを出力
Selector	入力ベクトル要素の選択や再配列
Signal Specification	信号の属性指定

表 9-8: Signals & Systems ライブラリプロック (Continued)

ブロック名	目的
Subsystem	複数システムをまとめて、別の一つのシステ ムに表現
Terminator	連結されていない出力端子を終結
Trigger	サブシステムに Trigger 端子を付加
Width	入力ベクトルの幅を出力

表 9-8: Signals & Systems ライブラリプロック (Continued)

目的 入力の絶対値を出力します。

ライブラリ Math

詳細 Abs ブロックは、入力の絶対値を出力として生成します。



プ

サポートされて ブロックは、double タイプの実数値または複素数値を受け入れ、double タイプの **いるデータタイ** 実数値出力を行います。

ダイ	ア	グ	゚ボ	ッ
クス				

Block Parameters: Abs	×
Abs	1
y = lul	
Parameters Saturate on integer overflow	
OK Cancel Help Apply	

Saturate on integer overflow (integer オーバーフローで飽和)

チェック(デフォルト)した場合、ブロックは、データタイプの最も負の値に相 当する符号付き整数値の入力を、データタイプの最も正の値に写像します。

- 8-bit integers の場合、-128 は 127 に写像されます。
- 16-bit integers の場合、-32768 は 32767 に写像されます。
- 32-bit integers の場合、-2147483648 は 2147483647 に写像されます。

チェックなしの場合、最も負の値に対応する符号付き整数値の入力要素に対する ブロックの挙動は、未定義です。

特性 直接フィードスルー あり サンプル時間サンプル 接続されるブロックからの継承 時間 スカラ拡張 N/A 次元化 可 ゼロクロッシング あり、ゼロ検出のため **目的** 代数ループを処理するもので、入力をゼロに設定します。

ライブラリ Math

詳細



Algebraic Constraint ブロックは、入力信号 f(z) をゼロに制約し、代数的な状態 z を出力します。ブロックは、入力時にゼロを生成するのに必要な値を出力します。出力は、いくつかのフィードバック経路を通じて入力に作用しなければなりません。これにより、インデックス1の微分代数方程式 (DAE) のシステムに対する代数方程式を指定することができます。

デフォルトでは、Initial guess(初期予測値)パラメータはゼロです。代数ループ ソルバの効率は、解の値に近い代数状態 z を Initial guess(初期予測値)に提供す ることによって改良することができます。

たとえば、下のモデルは、つぎの方程式を解きます。

 $z^{2} + z^{1} = 1$ $z^{2} - z^{1} = 1$

Display ブロックが示すとおり、解は $z_2 = 1$, $z_1 = 0$ です。



サポートされて Algebraic Constraint ブロックは、double タイプの実数値を受け入れ、出力します。 いるデータ

パラメータとダ イアログボック ス



Initial guess(初期予測值)

解の値の初期推定値。デフォルトは0。

特性

直接フィードスルー	あり
サンプル時間	接続されるブロックからの継承
スカラ拡張	不可
次元化	可
ゼロクロッシング	なし

目的 遊び(余裕)のあるシステムの挙動をモデル化します。

ライブラリ Nonlinear

詳細

×₩

Backlash ブロックは、入力の変化が出力に等しい変化を起こさせるシステムを実現します。しかし、入力の方向が変化するときは、入力の初期変化は出力に影響しません。システム内の端から端までの遊びの量は、*不感帯*と呼ばれます。不感帯は、出力に対して中央に位置します。つぎの図は、デフォルトの不感帯幅が1で、初期出力が0のブロックの初期状態を示しています。



遊びをもつシステムは、つぎの3つのモードのいずれかの状態にあります。

- 分離 このモードでは、入力は出力に影響せず、出力は一定のままです。
- 正方向の結合 このモードでは、入力は増加し(正の勾配をもちます)、出力は 入力から不感帯幅の半分を差し引いたものに等しくなります。
- 負方向の結合 このモードでは、入力は減少し(負の勾配をもちます)、出力は 入力に不感帯幅の半分を*加えた*ものに等しくなります。

初期入力が不感帯の外側にある場合、Initial output(初期出力)パラメータ値に よってプロックが正と負のいずれの方向に結合されているかが決まり、シミュ レーション開始時における出力は入力に不感帯幅の半分を加えたものか差し引い たものとなります。

たとえば、2つのギアの噛み合わせ状態をモデル化するために Backlash ブロック を使用することができます。入力と出力はいずれも一端にギアのある2つのシャ フトで、出力のシャフトは入力のシャフトで駆動されます。ギアの歯の間の余分 な空間が*遊びで*す。この空間の幅が Deadband width(不感帯の幅)パラメータで す。はじめにシステムが分離されている場合、出力(駆動されるギアの位置)は Initial output(初期出力)パラメータによって定義されます。 下の図は、初期入力が不感帯内にあるときのブロックの動作を示しています。最 初の図は、システムが分離モード(およびデフォルトのパラメータ値が変化して いない)状態での入力と出力の関係を示しています。



つぎの図は、入力が不感帯の端に到達して出力と噛み合ったときのブロックの状 態を示しています。出力はその前の値のままです。



最後の図は、入力と出力が噛み合った状態のときに入力の変化が出力にどのよう に影響するかを示しています。



入力の方向が反転すると、出力から分離されます。出力は、入力が不感帯の反対 の端に到達するか、その方向を再び反転して不感帯の同じ端に到達するまで一定 です。そして、前述と同様、入力が移動すると、出力が等しい量だけ移動しま す。

たとえば、不感帯幅が2で、初期出力が5の場合、シミュレーション開始時の出力 y は、つぎのようになります。

- 入力uが4と6の間の場合は5
- *u* < 4 の場合は u + 1
- u > 6 の場合は u 1

つぎのサンプルモデルとその後のプロットは、Backlash ブロックを通過する正弦 波の効果を示しています。



Backlash ブロックパラメータは、そのデフォルト値から変更しません(不感帯幅 は1で、初期出力は0です)。下図でプロットされている出力で、入力が(0.5 で) 不感帯の端に到達するまでは、Blacklash ブロック出力がゼロであることに注意 してください。つぎに、入力と出力が噛み合って、入力が(1.0 で)方向を変える まで、出力は入力にしたがって移動します。入力が0に到達すると、入力は不感 帯の反対側の端で再び出力と噛み合います。



サポートされて Backlash ブロックは、double タイプの実数値を受け入れ、出力します。 いるデータタイ プ

Backlash

パラメータとダ イアログボック ス

Block Paramet	ers: Backlasł	h			2
Model backla the system.	sh where the	e deadband w	idth specifie	es the amoun	t of play in
- Parameters -					
Deadband	width:				
1					
Initial outp	ut:				
0					
OK	0	Cancel	<u>H</u> elp		oly

Deadband width(不感帯の幅)

不感帯の幅。デフォルトは1。

Initial output(初期出力)

初期出力値。デフォルトは 0。

直接フィールドスルー あり サンプル時間 接続されるブロックから継承 スカラ拡張 可 次元化 可 ゼロクロッシング あり、上下のしきい値で結合を検出するため **目的** 連続システムに白色ノイズを導入します

ライブラリ Sources

詳細

╏╻┨

Band-Limited White Noise ブロックは、連続システムまたはハイブリッドシステム で使用する正規分布乱数を発生します。

このブロックと Random Number ブロックの主な違いは、Band-Limited White Noise ブロックがノイズの相関時間に関連した特定のサンプルレートで出力を発生することです。

理論的に、連続白色ノイズは0の相関時間と、平坦なパワースペクトル密度 (PSD)、および無限大の共分散をもっています。ノイズによる外乱がシステムの 固有帯域幅に比較して非常に小さな相関時間をもつ場合、白色ノイズは便利な理 論的近似ではありますが、物理システムが実際上白色ノイズによる外乱を受ける ことは決してありません。

Simulink では、システムの最短時間定数よりかなり小さな相関時間をもつ乱数列 を使って、白色ノイズの効果をシミュレーションすることができます。 Band-Limited White Noise ブロックは、そのような乱数列を発生します。ノイズの 相関時間は、ブロックのサンプルレートです。正確なシミュレーションのために は、システムの最高速のダイナミクスよりかなり小さな相関時間を使用してくだ さい。つぎのように指定することによって、適切な結果を得ることができます。

$$t_c \approx \frac{1}{100} \frac{2\pi}{f_{max}}$$

ここで、fmax はラジアン / 秒単位のシステムの帯域幅です。

ブロック実現で使用されるアルゴリズム

このノイズの正しい強度を発生するために、ノイズの共分散は、連続 PSD から 離散ノイズ共分散への暗黙の変換を反映するために、スケーリングされます。適 切なスケールファクタは、1/tc です。ここで、tc はノイズの相関時間です。この スケーリングは、近似白色ノイズに対する連続システムの応答が、真の白色ノイ ズを使用した場合にシステムがもつのと同じ共分散をもつようにします。このス ケーリングのため、Band-Limited White Noise ブロックからの信号の共分散は Noise power(ノイズパワー(強度))ダイアログボックスパラメータと同じではあ りません。このパラメータは、実際には白色ノイズの PSD の高さです。真の白 色ノイズの共分散は無限大であるのに対して、このブロックで使われる近似がも つ特性として、ブロック出力の共分散は、Noise power(ノイズパワー)を tc で除 算したものです。 サポートされて いるデータタイ プ

バラメータとダ

イアログボック

ス

 Block Parameters: Band-Limited White Noise
 X

 Continuous White Noise. (mask) [link]
 White noise for continuous (s-domain) systems. Band-limited using zero-order-hold.

 Parameters
 Noise power:

 [0.1]
 Sample time:

 [0.1]
 Seed:

 [23341]
 OK

Noise power(ノイズパワー)

白色ノイズの PSD の高さ。デフォルト値は 0.1。

Sample time(サンプル時間)

ノイズの相関時間。デフォルト値は0.1。

Seed($\vartheta - \vartheta$)

乱数発生器の初期値。デフォルト値は23341。

特性

離散
ノイズパワー (Noise power) および シード (Seed) パラ メータと出力について
可
なし

Band-Limited White Noise ブロックは、double タイプの実数値を出力します。

目的

符号なし整数信号のビットを論理マスク、反転、シフトします。

ライプラリ Math

詳細

> bitwise > AND 'FFFF' Bitwise Logical Operator は、符号なし整数信号のビットについて、論理マスク (AND, OR, XOR)、反転 (NOT), シフト (SHIFT_LEFT, SHIFT_RIGHT) 演算を行い ます。ブロックのパラメータダイアログを使って実行する演算を選択することが できます。Bitwise Logical Operator ブロックを使って、スカラの符号なし整数信 号と同様にベクトルについてもビット単位の演算を行うことができます。

マスク演算

Bitwise Logical Operator のマスク演算 (AND, OR, XOR) は、入力信号の各ビット を、マスクと呼ばれる定数オペランドの対応するビットと論理的に組み合わせま す。マスクの値と論理演算をブロックのパラメータダイアログによって指定しま す。マスクと論理演算は、出力信号の各ビットの値を以下のように決定します。

演算	マスクビット	入力ビット	出力ビット
AND	1	1	1
	1	0	0
	0	1	0
	0	0	0
OR	1	1	1
	0	1	1
	1	0	1
	0	0	0
XOR	1	1	0
	1	0	1
	0	1	1
	0	0	0

Bitwise Operator ブロックでは、信号とマスクの両方の配列が利用できます。一般的には、マスクは入力信号と同じ次元をもっていなければなりません。つまり、

5 × 4 の入力信号には 5 × 4 のマスクが必要です。ブロックは、マスクの各要素 を対応する入力要素に割り当てます。つぎの例外にも、入力とマスクは同じ次元 にしなければならないという一般的なルールが当てはまります。

- 入力がスカラでマスクが配列の場合、ブロックは、各マスク要素を入力に対応させた結果構成される配列を出力します。
- 入力が配列でマスクがスカラの場合、ブロックは、マスクを各入力要素に対応させた結果構成される配列を出力します。
- 入力が1-D配列(つまりベクトル)の場合、マスクは1行か1列のベクトルにする ことができます。

マスキング演算を選択する際に、ブロックのパラメータダイアログの Second operand(第2オペランド)フィールドを使って単数または複数のマスクを指定し ます。スカラ、ベクトル、セル配列を評価する任意の MATLAB 表現を入力する ことができます。マスク表現で文字列を用いると、16 進数を指定します(例, 'FFFF')。

必要ならば、ブロックはマスクの高次のビットを打ち切って、入力信号のデータ タイプのワードサイズに合わせます。たとえば、マスクの値を 'FF00' と指定し、 入力信号のタイプが uint8 であるとします。ブロックは、指定した値を '00' に打 ち切ります。

ベクトルを使って16進のマスクを指定することができますが、この方法の思わ ぬ落とし穴に気を付けてください。たとえば、MATLAB表現['00' 'FF']は、2つ の文字列ではなく、単一の文字列'FF00'を表わします。同様に、表現['FFFF'; '0000']は、2つの文字列を表わしますが、表現['FFFF'; '00']は不正であり MATLABはエラーを発生します。これらの間違いは、マスクに対して常にセル 配列を使って16進数値を指定するか、10進数と16進数を混在させることに よって防ぐことができます。たとえば、つぎのモデル



は、セル配列({'F0''0F'})を用いて2要素入力ベクトルのマスクに対する16進値 を指定します。

反転演算

Bitwise Logical Operator の NOT 演算は、入力信号のビットを反転します。特に、 入力信号について1の補数演算を行い、対応する入力ビットが0である場合は各 ビットが1である出力信号を生成し、逆の場合も同様です。

シフト演算

プ

ス

パラメータとダ イアログボック

Bitwise Logical Operator のシフト演算 SHIFT LEFT および SHIFT RIGHT は、入力 信号のビットを左または右にシフトして、出力信号を生成します。シフトの量は ブロックのパラメータダイアログの Second operand(第2オペランド)フィール ドで指定します。入力信号のワードサイズよりも大きいシフト量を指定する場合 は、ブロックは入力のワードサイズをシフト量として利用し、その結果出力はゼ 口信号になります。

サポートされて Bitwise Logical Operator は、符号なし整数のデータタイプ、uint8, uint16, uint32の 実数値入力を受け入れます。ベクトル入力のすべての要素は、同じデータタイプ いるデータタイ でなければなりません。出力信号は、入力と同じデータタイプです。

Block Parameters: Bitwise Logical Operator				
BitwiseOperator (mask) (link)				
Perform a bitwise operation on uint8, uint16 or uint32 input port data with values from the Second operand parameter. Hex values can be entered as strings, e.g., 'FF00'.				
Parameters				
Bitwise operator:	AND			
Second operand:				
'FFFF'				
OK	Cancel <u>H</u> elp Apply			

Bitwise operator(ビット毎演算)

入力信号に適用するビット単位の演算子を指定

Second operand(第2オペランド)

マスク演算に対するマスクオペランドとシフト操作に対するシフト量を指定 します。スカラ、ベクトル、セル配列を評価する任意の MATLAB 表現を入 力することができます。ブロック入力がベクトルである場合は、ブロックは 各値を対応する入力の要素に適用します。入力がスカラである場合は、ブ ロックは、対応するパラメータ値を入力に適用した結果を要素とするベクト ルを出力します。

特性	サンプル時間	接続されるブロックから継承
	スカラ拡張	入力と Second operand パラメータについて
	次元化	可
	状態	なし
	ゼロクロッシング	なし
	直接フィードスルー	あり
目的 入力バスから入ってくる信号を選択します。

れ、出力します。

ライブラリ Signals & Systems

詳細

×

Bus Selector ブロックは、Mux ブロックまたは他の Bus Selector ブロックからの入 力を受け入れます。このブロックは、1つの入力端子をもっています。出力端子 の数は、Muxed output(出力信号のベクトル化)チェックボックスによって指定 されます。Muxed output(出力信号のベクトル化)をチェックしている場合、信 号は出力端子で合成され、出力端子はただ1つしか存在しないことになります。 これに対して、チェックされていない場合、選択されたそれぞれの信号に対して 1出力端子が割り当てられることになります。

注意 Simulink は、Simulink ライブラリからモデルにコピーしたとき、Bus Selector ブロックの名前を非表示にします。

Bus Selector ブロックは、任意のデータタイプの実数値または複素数値を受け入

サポートされて いるデータタイ プ パラメータとダ イアログボック ス

🚮 Block Parameters: Bus Selector 📃 🗖 🔀		
Bus Selector This block accepts input from a Mux or Bus Selector block. The left listbox shows the signals in the input bus. Use the Select button to select the output signals. The right listbox shows the selections. Use the Up, Down, or Remove button to reorder the selections. Check "Muxed output' to multiplex the output.		
Signals in the bus	Selected signals Up position velocity Remove	
Refresh Select >>	Muxed output	
ОК	Cancel Help Apply	

Signals in the bus(バス内の信号)

Signals in the bus リストボックスは、入力バスに入力されている信号を表示 します。選択 >> ボタンによって、Signals in the bus リストボックスから出力 信号を選択できます。

Selected signals(選択された信号)

Selected signals リストボックスは、出力信号を表示します。ユーザは、Up(アップ)、Down(ダウン)とRemove(削除)ボタンによって信号を並べ替え ることができます。信号の順番が変更されたときも、端子の連結性は維持さ れます。

Selected signals リストボックスに表示されている出力信号が Bus Selector ブロックに対する入力となっていない場合、信号の名称欄には ??? の文字が表示されます。

出力端子上の信号に付けられるラベルは、Muxed output チェックボックス をチェックすると、このブロックによって自動的に設定されます。このラベ ルを変更しようとすると、Bus Selector ブロックの出力に繋がっているライ ンの信号ラベルを変更することができない旨を知らせるエラーメッセージが 表れます。 目的 周波数が増加する正弦波を発生します。

ライブラリ Sources

詳細



Chirp Signal ブロックは、周波数が時間に比例して増加する正弦波を発生します。 このブロックは、非線形システムのスペクトル解析のために使用することができ ます。ブロックは、スカラまたはベクトルの出力を発生します。

パラメータの Initial frequency(初期周波数)、Target time(ターゲット時間)、お よび Frequency at target time(ターゲット時間の周波数)は、ブロックの出力を決 定します。これらの変数の一部やすべてをスカラや配列に指定することができま す。配列として指定されるパラメータは、すべて同じ次元でなければなりませ ん。ブロックは、スカラーパラメータを、同じ次元をもつ配列パラメータに拡張 します。ブロックの出力は、Interpret vector parameters as 1-D(ベクトルパラ メータを 1-D として解釈)オプションが選択されているとき以外はパラメータと 同じ次元です。このオプションが選択され、パラメータが行ベクトルや列ベクト ルの場合、ブロックはベクトル(1-D 配列)信号を出力します。

Chirp Signal ブロックは、double タイプの実数値信号を出力します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Chirp Signal 🛛 🗙
_ chirp (mask) (link)
Chirp Signal. (Sine wave with increasing frequency)
Parameters Initial frequency (Hz):
Target time (secs):
100
Frequency at target time (Hz):
1
☑ Interpret vectors parameters as 1-D
OK Cancel Help Apply

Initial frequency(初期周波数)

スカラまたはベクトルの値として指定した信号の初期周波数。デフォルトは 0.1 Hz。

Target time(ターゲット時間)

周波数が Frequency at target time パラメータ値に到達する時間で、スカラまたはベクトル値。この時間後も、周波数は引き続き同じ割合で変化します。 デフォルトは 100 秒。

Frequency at target time(ターゲット時間の周波数)

ターゲット時間における信号の周波数で、スカラまたはベクトル値。デフォルトは1Hz。

Interpret vector parameters as 1-D(ベクトルパラメータを 1-D として解釈)

このオプションを選択すると、Initial frequency、Target time および Frequency at target time のパラメータの列または行マトリクスの値が行要素 あるいは列要素のベクトル出力となります。

サンプル時間	連続
スカラ拡張	パラメータについて
次元化	可
ゼロクロッシング	なし

目的 シミュレーション時間を表示、設定します。

ライブラリ Sources

詳細 Clock ブロックは、各シミュレーションステップでカレントのシミュレーション 時間を出力します。このブロックは、シミュレーション時間を必要とするような 他のブロックにとっては、便利なブロックとなります。

> 離散システム内で、カレント時間を必要とする場合は、Digital Clock ブロックを 用いてください。

Clock ブロックは、double タイプの実数値信号を出力します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Clock	×
Output the current simulation time.	
Parameters	
🔽 Display time	
Decimation:	
10	
OK Cancel <u>H</u> elp <u>Apply</u>	

Display time(表示時間)

Display time チェックボックスを用いて、カレントのシミュレーション時間 を Clock ブロックアイコンの中に表示させることができます。

Decimation(間引き)

Decimation パラメータの値は、時計が更新時にどれだけ更新されるかを指定 します。このパラメータには正の整数を入力できます。たとえば、間引き = 1000を指定すると、1ミリ秒の固定積分ステップあたり、時計は1秒、2秒 というように1秒ずつ更新されていきます。このパラメータが非ゼロの場 合、正確に時計が更新されるようにするために、固定積分ステップシミュ レーションを使用するしなければならないことに注意してください。

特性 サンプル時間 連続

スカラ拡張 N/A

次元化 不可 ゼロクロッシング なし **目的** 真理値表を実現します。

ライブラリ Math

詳細

>[:::]>

Combinatorial Logic ブロックは、プログラミング可能な論理配列 (PLAs)、論理回路、決定表、その他のプール式をモデル化するための、標準の真理値表を実現します。このブロックは Memory ブロックと組み合わせて使用することにより、有限状態マシンやフリップフロップを実現することができます。

Truth table パラメータとして、すべての可能なブロック出力を定義する行列を指定します。行列の各行には、異なる入力要素の組み合わせが含まれています。入力のすべての組み合わせに対して、出力を指定する必要があります。列数は、ブロックの出力数です。

入力数と行数の関係は、つぎのとおりです。

行数 = $2^{(\lambda)}$ (入力数)

Simulink は、入力ベクトル要素から行のインデックスを計算することによって、 行列の1つの行を出力します。Simulink は、入力ベクトルのゼロ要素が0で、非 ゼロ要素が1である2進数を作成することによってインデックスを計算し、結果 に1を加算します。m要素の入力ベクトルuの場合には、つぎのようになりま す。

行インデックス = $1 + u(m)*2^{0} + u(m-1)*2^{1} + ... + u(1)*2^{m-1}$

2入力の AND 関数の例

この例では、2つの入力要素がいずれも1のときに1を出力し、そうでないとき に0を出力する2入力のAND 関数を作成します。この関数を実現するには、 Truth table パラメータ値を[0;0;0;1]と指定します。Combinatorial Logic ブロック に対して入力と出力を提供するモデルの一部は、つぎのようになります。



下の表は、各出力を生成する入力の組み合わせを示しています。"入力1"のラベルの付いた入力信号は、入力1のラベルの付いた表の列に対応します。同様に、 "入力2"のラベルの付いた入力信号は、入力2の列に対応します。これらの値の 組み合わせによって、表の出力列のどの行がブロック出力として渡されるかが決まります。

たとえば、入力ベクトルが [10] の場合、	入力は第3行(2 ¹ *1+1)を参照します。
したがって、出力値は0です。	

行	入力1	入力 2	出力
1	0	0	0
2	0	1	0
3	1	0	0
4	1	1	1

回路の例

このサンプル回路には3つの入力があります。加算される2つのビット(aとb) とキャリ-インビット(c)です。出力は、キャリ-アウトビット(c)と加算ビット(s)の2つです。以下に、真理値表とこの回路に対する入力値の組み合わせに 関連した出力を示します。

入力		出	Ъ	
а	b	с	c'	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

この加算器を Combinatorial Logic ブロックで実現するには、Truth table パラメー タとして、 $c' \ge s$ の列で作られる 8 行 2 列の行列を入力します。

順序回路(すなわち、状態をもつ回路)もブロックの状態に対する入力を追加 し、ブロックの出力をこの状態入力にフィードバックすることによって、 Combinatorial Logic ブロックで実現することができます。

サポートされて いるデータタイ プ

Combinatorial Logic ブロックが受け入れる信号のタイプは、Simulinkの Boolean logic 信号オプション(4-49ページの"厳密な Boolean 型の検査を有効にする"を参 照)を選択するかどうかに依存します。このオプションが選択されている場合、 ブロックは、boolean または double タイプの実数信号を受け入れます。真理表 は、任意のデータタイプの Boolean 値(0 あるいは1)をとります。真理表が非 Boolean 値を含んでいる場合、表のデータタイプは double にしなければなりませ ん。出力タイプは、入力が boolean で真理表が非 boolean 値の場合でブロックが double を出力する場合を除いて、入力と同じタイプになります。Boolean 互換性 モードが選択されていない場合、Combinatorial Logic ブロックは、boolean タイプ の信号だけを受け入れます。ブロックは、真理表が double タイプの非 Boolean 値 を含む場合、double を出力します。そうでない場合、出力は boolean です。

パラメーク	タとダ
イアログス	ドック
ス	

Block Parameters: Cor	mbinatorial Logic		×
Combinatorial Logic			
Look up the elements of the input vector (treated as boolean values) in the truth table and outputs the corresponding row of the 'Truth table' parameter. The input side of the truth table is implicit.			
- Parameters			
Truth table:			
[0 0;0 1;0 1;1 0;0	1;1 0;1 0;1 1]		
OK	Cancel	<u>H</u> elp	

Truth table(真理值表)

出力の行列。各列は出力ベクトルの要素に対応し、各行は真理値表の行に対応します。

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	不可
次元化	可、出力幅は Truth table パラメータの列数
ゼロクロッシング	なし

目的 複素数入力信号の大きさと位相角を出力します。

ライブラリ Math

Complex to Magnitude-Angle ブロックは、double タイプの複素数表示の信号を受 詳細 け入れます。そして、Output パラメータの設定によって、入力信号の大きさ及 び/または、位相角を出力します。出力される値は、double タイプの実数値で す。複素数信号のベクトルが入力される場合、出力信号もベクトルとなります。 大きさの信号ベクトルは、対応する複素数入力要素の大きさを示しています。角 度出力は、同様に入力された要素の角度を示しています。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

上の詳細を参照してください。

Block Parameters:	×
Complex to Magnitude-Angle Compute magnitude and/or radian phase angle of the input.	
Parameters Output: MagnitudeAndAngle	
OK Cancel Help	Apply

Output(出力)

このブロックの出力を決定します。選択できる設定値は、 MagnitudeAndAngle(入力信号の大きさと位相角(ラディアン表示)を出力)、Magnitude (入力信号の大きさ)、Angle (入力の位相角をラディアン表示 で出力)のいずれかです。

特性

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	不可
次元化	可
ゼロクロッシング	不可

目的 複素数入力信号を構成する実数部と虚数部を出力します。

ライプラリ Math

詳細



プ

Complex to Real-Imag ブロックは、任意のタイプの複素数値信号を入力として受け入れます。そして、Output(出力)パラメータの設定によって、入力信号の実数部及び/または虚数部を出力します。実数出力は、複素数入力と同じデータタイプです。入力は、複素数信号からなる配列(ベクトルまたは行列)でもかまいません。この場合、出力信号は同じ大きさの配列です。実数配列には、対応する複素数入力要素の実部が含まれます。虚数出力は、同様に入力要素の虚部が含まれます。

上の詳細を参照してください。

パラメータとダ イアログボック ス

サポートされて

いるデータタイ

Block Parameters:				×
Complex to Real-ima Output the real and/o	ag r imaginary comp	onents of the inpu	ut.	
Parameters				
Output: RealAnd	mag			
ОК	Cancel	<u>H</u> elp	L L	Apply

Output(出力)

レフロ

古位う、

このブロックの出力を決定します。つぎの値、RealAndImag(入力信号の実部と虚部)、Real(入力信号の実部)、Imag(入力信号の虚部)から選択します。

特性

且按ノイートスルー	のリ
サンプル時間	接続されるブロックから継承
スカラ拡張	不可
次元化	可
ゼロクロッシング	なし

もい

目的 指定したライブラリから選択した特定のブロックにアクセスして実行します。

ライブラリ Signals & Systems

詳細



Configurable Subsystem ブロックは、指定したブロックライブラリに含まれるあ るブロックにアクセスして実行します。Configurable Subsystem ダイアログボッ クスにより、ユーザはどのブロックにアクセスして実行するかを指定し、そのブ ロックのパラメータの数値を設定することができます。

Configurable Subsystem ブロックは、様々な設計図の集合に相当するようなモデ ル作成を単純化してくれます。たとえば、エンジンを自由に選ぶことができるよ うな自動車のモデル化に着手しようとしていると仮定しましょう。そのような設 計図をモデル化するために、ユーザは最初に自動車を作る際に利用可能なエンジ ンタイプのモデルからなるライブラリを作成します。このとき、エンジンの選択 にあたる部分を Configurable Subsystem ブロックで定義することにより、車のモ デル上でこのブロックを用いることができます。基本的な自動車のデザインの一 部を特に変更したバリエーションをモデル化しようとするならば、ユーザは、エ ンジンタイプの選択のみを行えば良いわけですが、ここで、Configurable Subsystem ブロックのダイアログを活用することができます。

Configurable Subsystem ブロックの外観は、現在設定しているブロックに応じて 変化します。最初の段階で、Configurable Subsystem ブロックは、何も表示して いません。この時点では、Configurable Subsystem ブロックは、端子をもたず、 左記のようなアイコンを表示するだけです。ユーザがライブラリとブロックを選 択した時点で、Configurable Subsystem は選択したライブラリの入力端子と出力端 子に対応するアイコンと入出力端子を表示します。

Simulink は、ライブラリ端子を Configurable Subystem ブロック端子に割り当てる とき、つぎのルールに従います。

- ライブラリ内のユニークに名付けられた入力/出力端子をConfigurable Subystem ブロック上に同じ名前の別々な入力/出力端子に割り当てます。
- ライブラリ内のすべて同じに名付けられた入力 / 出力端子を Configurable Subystem ブロック上の同じ入力 / 出力端子に割り当てます。
- Terminator/Ground ブロックをもつ現在選択したライブラリブロックにより使われていない入力 / 出力端子を切断します。

この写像は、Configurable Subsystem ブロックに再度結合させないで、 Configurable Subsystem ブロックにより表現されるライブラリブロックを変更さ せることができます。 たとえば、表示されているライブラリが2つのブロックA、Bを含み、ブロック Aはa,b,cとラベル付けされている入力端子をもち、dとラベル付けされている 出力端子をもっているとします。一方、ブロックBはaおよびbとラベル付けさ れている入力端子と、eとラベル付けされた出力端子をもっているとします。こ のライブラリに基づく Configurable Subsystem ブロックは、以下の図に示すよう に、a,b,cとラベル付けられた3つの入力端子をもち、dおよびeとラベル付けら れた2つの出力端子をもちます。



この例で、Configurable Subystem ブロックの端子 a は、どのブロックが選択されていても、選択されたライブラリブロックの端子に接続されます。一方、 Configurable Subsystem ブロックの端子 c は、ライブラリブロック A が選択されている場合にのみ機能します。そうでない場合は、終了します。

注意 Configurable Subsystem ブロックは、triggered and enabled サプシステムの trigger や enable 端子のような非 I/O 端子に対応する端子を提供しません。そのた め、そのような端子をもつブロックを表わすため、Configurable Subsystem ブロッ クを直接使うことはできません。しかし、非 I/O 端子に結合された入出力端子を もつサブシステムブロックにそのようなブロックを含めることにより、間接的に 行うことができます。

configurable subsytem の作成方法

- configurable subsystem を構成する様々なブロックを含んだライブラリを作成します。
- ライブラリを Configurable Subsystem に設定します。作成するためには、 Simulink の Signals and Systems ライブラリの Configurable Subsystem のコピー を、前のステップでユーザが作成したライブラリの中にドラッグしてください。
- ブロックをダブルクリックして Configurable Subsystem ブロックダイアログを 表示させます。

4 作成した configurable subsystems を構成するブロックにチェックをします。

5 ダイアログを閉じます。

ライブラリを保存します。

サポートされて Configurable Subsystem ブロックは、カレントに表示されているブロックが受け いるデータタイ 入れ、出力するのと同じタイプの信号を受け入れ、出力します。 プ

パラメータとダ イアログボック ス

Configurable Subsystem のダイアログボックスでは、Configurable Subsystem ブ ロックが現在ライブラリを表示しているのかどうか、そして、ブロックを表示し ている場合、どのブロックを表示しているのか、によって変化します。初期状態 では、Configurable Subsystem ブロックは何も表示していません。そのため、ダ イアログボックスには空の Library name パラメータしか表示しません。

- 4). (Configuration d	alog : Con	figurable Subsystem	_ 🗆 ×
F F S	Configuration d st of block choices Block name Pulse Generator Random Number Sine Wave	Member	fgurable Subsystem Port information Inports Outports Port names Out2 Out1	Up Down
		ок	Cancel Help	Apply

List of block choices

configurable subsystem の構成メンバーに加えたいブロックにチェックをします。ブロックにはユーザ定義サブシステムを加えることができます。

Port information

メンバーブロックの出力端子と入力端子のリストです。複数端子の場合、Up や Down ボタンを押して選択した端子の配置を入換えることができます。

注意 ライブラリ内でブロックや端子追加したり削除した場合、ライブラリを使用している Configurable Subsystem を作り直さなければなりません。

つぎの図は、Configurable Subystem ブロックのダイアログボックスです。

bsystem ect the settings	for the subsystem block.
rameters V Show port I	labels
Block choice:	Pulse Generator
Read/Write pe	rmissions: ReadWrite
Name of error of	callback function:
🗖 Treat as at	omic unit
RT₩ system c	ode: Auto 💌
RTW function	name options: Auto
Function name	
RTW file name	options: Auto

Block choice

特性

この Configurable Subystem ブロックが現在表現しているブロック。このメ ニューには、ユーザの configurable subsystem ライブラリ内のすべてのブロッ クのリストが表示されます。

Block choice の下のパラメータは、サブシステムの属性に関するものです。詳細は、Subsystem block リファレンスのページをご覧ください。

Configurable Subsystem ブロックは、現在表現しているブロックの特徴を保持しま す。ブロックをダブルクリックすると、現在表現しているブロックのダイアログ ボックスが開きます。

Constant

目的 定数値を発生します。

ライブラリ Sources

詳細



Constant ブロックは、時間に依存しない指定した実数値または複素数値を発生し ます。ブロックは、Constant value パラメータの長さによって、また Interpret vector parameters as 1-D パラメータの設定によって、スカラ、ベクトル、行列の 出力を生成します。Interpret vector parameters as 1-D パラメータが選択され、 Constant value パラメータが列または行の行列である場合は、出力は、要素がパ ラメータの要素である1次元配列(ベクトル)です。そうでない場合は、出力 は、パラメータと同じ大きさで、要素がパラメータの要素である2次元配列(行 列)です。

サポートされて いるデータタイ プ

Constant ブロックは、数値タイプ(複素数または実数)とデータタイプがブロックの Constant value パラメータと同じである信号を出力します。

パラメータとダ イアログボック ス

Block Parameters: Constant	×	
Constant		
Output the constant specified by the 'Constant value' parameter. If 'Constant value' is a vector and 'Interpret vector parameters as 1-D' is on, treat the constant value as a 1-D array. Otherwise, output a matrix with the same dimensions as the constant value.		
Parameters Constant value:		
0		
☑ Interpret vector parameters as 1-D		
OK Cancel Help Apply		

Constant value

ブロックの出力

Interpret vector parameters as 1-D

選択された場合、Constant value パラメータに対する列または行の行列値は、 要素が行または列の要素であるベクトル出力になります。

特性	サンプル時間	定数
	スカラ拡張	不可

次元化 可 ゼロクロッシング なし **目的** 零点で不連続性を、それ以外では線形ゲインをモデル化します。

ライブラリ Nonlinear

詳細

Coulomb and Viscous Friction ブロックは、クーロン(スタティック)摩擦と粘性(ダイナミック)摩擦をモデル化します。ブロックは零点では不連続性をモデル化 し、それ以外では線形ゲインをモデル化します。オフセットはクーロン摩擦に対 応し、ゲインは粘性摩擦に対応します。ブロックはつぎのように実現されます。

y = sign(u) * (Gain * abs(u) + Offset)

ここで、yは出力、uは入力、Gain と Offset はブロックパラメータです。 ブロックは、1 つの入力を受け入れ、1 つの出力を生成します。

Coulomb and Viscous Friction ブロックは、double タイプの実数信号を受け入れ、

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Coulomb & Viscous Friction 🛛 🛛 💌
Coulombic and Viscous Friction (mask) (link)
A discontinuity offset at zero models coulomb friction. Linear gain models viscous friction. y = sign(x) * (Gain * abs(x) + Offset)
Parameters
Coulomb friction value (Offset):
[1 3 2 0]
Coefficient of viscous friction (Gain):
1
OK Cancel <u>H</u> elp <u>Apply</u>

Coulomb friction value

出力します。

すべての入力値に適用されるオフセット。デフォルトは [1320]。

Coefficient of viscous friction

非ゼロ入力点での信号のゲイン。デフォルトは1。

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	不可

次元化 可 ゼロクロッシング あり、スタティックな摩擦を超える瞬間 目的 データストアを定義します。

ライブラリ Signals & Systems

Data Store Memory ブロックは、名前付き共有データストアを定義し初期化しま す。共有データストアとは、Data Store Read および Data Store Write ブロックが利 用できるメモリ領域です。

各データストアは、Data Store Memory ブロックで定義しなければなりません。 データストアを定義する Data Store Memory ブロックの位置によって、データス トアにアクセスできる Data Store Read および Data Store Write ブロックが決まり ます。

- Data Store Memory ブロックが トップレベルシステム内にある場合、モデル内の任意の位置にある Data Store Read および Data Store Write ブロックにより、データストアにアクセスすることができます。
- Data Store Memory ブロックが サブシステム内にある場合、同じサブシステム内、またはモデル階層において、そのサブシステムより下の任意のサプシステム内の Data Store Read および Data Store Write ブロックにより、データストアにアクセスすることができます。

データストアは、Initial value パラメータに値を指定することによって初期化し ます。値のサイズでデータストアのサイズが決まります。Data Store Write ブロッ クが同じデータ量を書き込まなければエラーとなります。

Data Store Memory ブロックは、double タイプの実数信号をストアします。

サポートされて いるデータタイ プ

詳細

パラメー	-タとダ
イアログ	ブボック
ス	

Block Parameters: Data Store Memory	×
DataStoreMemory	
Define a memory region for use by the Data Store Read and Data Store Write blocks.	
Parameters	
Data store name:	
Initial value:	
0	
Interpret vector parameters as 1-D	
OK Cancel <u>H</u> elp <u>Apply</u>	

Data store name

定義されるデータストア名。デフォルトは A。

Initial value

データストアの初期値。デフォルト値は0。

Interpret vector parameters as 1-D

これを選択すると、Initial value パラメータの列マトリクスあるいは行マトリ クスの値は、ベクトルの要素が行ベクトルや列ベクトルの要素の値に等しい 1-次元配列(ベクトル)にデータストアを初期化します。

サンプル時間	N/A
次元化	可

目的 データをデータストアから読み込みます。

ライブラリ Signals & Systems

詳細

A

Data Store Read ブロックは、データを名前付きのデータストアから読み込んで、 そのデータを出力として渡します。データは、あらかじめ Data Store Memory ブ ロックで初期化されて、(おそらく)Data Store Write ブロックでデータストアに 書き込まれたものです。

データを読み込むためのデータストアは、そのデータストアを定義する Data Store Memory ブロックの位置で決まります。詳細は、9-44 ページの Data Store Memory を参照してください。

Data Store Read ブロックは、double タイプの実数信号を出力します。

同じデータストアから、複数の Data Store Read ブロックを読み込むことができます。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Data Store Read	×
C Data Store Read	
Read values from specified data store.	
Parameters	5
Data store name:	
A	
Sample time:	
4	
OK Cancel <u>H</u> elp <u>A</u> pply	

Data store name

このブロックがデータを読み込むデータストア名。

Sample time

ブロックがデータストアに書き込むタイミングを制御するサンプル時間。デフォルトは-1 で、サンプル時間は継承されます。

サンプル時間 連続または離散

次元化

可

目的 データをデータストアに書き出します。

ライブラリ Signals & Systems

詳細

×A

Data Store Write ブロックは、ブロック入力を名前付きのデータストアに書き込みます。

Data Store Write ブロックが実行する各書き出し作業で、データストアは上書きされ、前の内容が置き換えられます。

このブロックが書き込みを行うデータストアは、そのデータストアを定義する Data Store Memory ブロックの位置で決まります。詳細は、9-44 ページの Data Store Memory を参照してください。データストアのサイズは、データストアを定義お よび初期化する Data Store Memory ブロックによって設定されます。そのデータ ストアに書き込む各 Data Store Write ブロックは、同じ量のデータを書き込まな ければなりません。

同じデータストアに、複数の Data Store Write ブロックから書き込むことができ ます。しかし、2 つの Data Store Write ブロックが、同じシミュレーションステッ プの同じデータストアに書き込もうとすると、結果は予測できなくなります。

Data Store Write ブロックは、double タイプの実数信号を受け入れます。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Data Store Write	×
Data Store Write	
Write values to specified data store.	
Parameters	
Data store name:	
A	
Sample time:	
-1	
OK Cancel <u>H</u> elp <u>Apply</u>	

Data store name

このブロックがデータを書き込むデータストア名

Sample time

ブロックがデータストアに書き込む時間を制御するサンプル時間。デフォル トは -1 で、サンプル時間は継承されます。
 特性
 サンプル時間
 連続または離散

 次元化
 可

目的 入力信号を指定したデータタイプに変換します。

ライブラリ Signals & Systems

詳細

プ



サポートされて

いるデータタイ

Data Type Conversion ブロックは、このブロックの Data type パラメータで指定さ れたデータタイプに入力信号を変換します。実数値または複素数値信号を入力す ることができます。実数信号が入力された場合、出力も実数値となります。複素 数信号が入力された場合、出力も複素数となります。

上の詳細を参照してください。

パラメータとダ イアログボック ス

Block Parameters: Da Data Type Convers Convert input signal	a <mark>ta Type Convers</mark> sion to specified data	ion type.	×
Parameters Data type:	ouble		
Saturate on	integer overflow Cancel	<u>H</u> elp	Apply

Data type

入力信号を変換するタイプを指定します。autoオプションは、入力信号を Data Type Conversion ブロックの出力端子が接続されている入力端子に必要と されるタイプに変換します。

Saturate on integer overflow

このパラメータは、整数出力に対してのみ作用します。このオプションを選 択すると、Data Type Conversion ブロックの出力が整数オーバブローしている ことを示せます。特に、出力データタイプが整数タイプの場合、出力タイプ により表現される最大値かまたは変換された出力に関して、絶対値の意味で 小さいほうの値をブロック出力とします。オプションを選択しない場合、 Simulink は、Simulation Parameters ダイアログボックスの Diagnostics ページ の Data overflow イベントオプションで指定された挙動を示します (5-25 ペー ジの "Diagnostics ページ"を参照)。

直接フィードスルー	Yes
サンプル時間	接続されるブロックから継承
スカラ拡張	パラメータによる N/A
次元化	可
ゼロクロッシング	あり、極限に到達する瞬間を検出する。

目的 ゼロ出力の領域を生成します。

ライブラリ Nonlinear

詳細



Dead Zone ブロックは、指定領域内に不感帯と呼ばれるゼロ出力を生成します。 不感帯の下限および上限は、Start of dead zone および End of dead zone パラメー タとして指定されます。ブロック出力は、入力と不感帯によって決まります。

- 入力が不感帯内にある(下限より大きく上限より小さい場合)。出力はゼロです。
- 入力が上限以上の場合、出力は入力から上限を差し引いたものになります。
- 入力が下限以下の場合、出力は入力から下限を差し引いたものになります。

つぎのサンプルモデルは、入力に下限-0.5と上限0.5の正弦波を使用します。



つぎのプロットは、正弦波について Dead Zone ブロックの効果を示しています。 入力(正弦波)が-0.5 と 0.5 の間にある場合、出力はゼロです。



サポートされて Dead Zone ブロックでは、任意のデータタイプの実数信号の入出力が可能です。 いるデータタイ プ

パラメータとダ イアログボック ス



Start of dead zone

不感帯の下限。デフォルトは-0.5。

End of dead zone

不感帯の上限。デフォルトは0.5。

特性	直接フィードスルー	あり
	サンプル時間	接続されるブロックから継承
	スカラ拡張	パラメータによる
	次元化	可
	ゼロクロッシング	あり、限界に到達する瞬間を検出するため

目的 ベクトル信号を複数の出力信号に分離します。

ライブラリ Signals & Systems

詳細

,

Demux ブロックは、入力信号の要素を分離し、別々の信号として出力します。 このブロックへの入力は、ベクトル(1-次元配列)信号やバス信号(詳細は 4-32 ページの ÉoÉXèMçÜ を参照)です。Number of outputs パラメータによって、数と オプションで各出力端子の次元を指定することができます。出力の次元を指定し ない場合は、ブロックが出力の次元を決めます。

Demux ブロックは、Bus selection mode パラメータの選択に依存して、ベクトルか バスかのどちらかの選択モードで働きます。2つのモードは、入力信号のタイプ が異なります。ベクトルモードはベクトルのような信号、つまり、スカラー(1 要素配列)かベクトル(1次元配列)、あるいは1列あるいは1行ベクトル(1行あ るいは1列の2次元配列)の信号だけを入力できます。

Demux ブロックの Number of outputs パラメータで、ブロックが動作するモード に依存したブロック出力の数と次元を決定します。

ベクトルモードで出力数を指定

ベクトルモードでは、パラメータの値を、出力の数を指定したスカラーやブロックの出力端子の幅を指定した要素からなるベクトルにすることができます。ブロックの出力のサイズは、入力信号のサイズと Number of outputs パラメータの値から決められます。

つぎの表は、幅nの入力ベクトルの出力を決める方法をまとめたものです。

パラメータ値	プロック出力	注釈
p = n	p スカラー信号	たとえば、入力が3要素ベク トルで3つの出力を指定と、 ブロックは、3つのスカラー 信号を出力します。
p > n	エラー	
p < n $n \mod p = 0$	それぞれが n/p 要素 をもった p ベクトル 信号	入力が6要素ベクトルで3出 力とすると、ブロックは、3 つの2要素ベクトルを出力し ます。

Demux

パラメータ値	プロック出力	注釈
p < n n mod $p = m$	(n/p)+1 要素をもった m ベクトル 信号と n/ p 要素をもった p-m 信号	入力が5要素ベクトルで、3 つの出力を指定すると、ブ ロックは2つの2要素ベクト ル信号と1つのスカラー信号 を出力します。
$\label{eq:product} \begin{split} & [p_1 \ p_2 \ \cdots \ p_m] \\ & p_1 + p_2 + \ldots + p_m = n \\ & p_i > 0 \end{split}$	幅が P1, P2, … P _m の m ベクトル信号	入力が 5 要素ベクトルで出力 として [3,2] を指定すると、 ブロックは 1 つの端子に入力 要素のうちの 3 つを、そし て、もう一方の端子に残りの 2 要素を出力します。
$[p_1 \ p_2 \ \ p_m]$ $p_1+p_2++p_m=n$ some or all $p_i = -1$	m ベクトル信号	_{Pi} がゼロより大きい場合、対 応する出力は幅 _{Pi} です。 _{Pi} が -1 の場合、対応する出力の 幅は動的なサイズです。
$[p_1 \ p_2 \ \ p_m]$ $p_1+p_2++p_m!=n$ $p_i = > 0$	エラー	

出力の数は、入力要素の数より少ない数を指定できることに注意して下さい。こ の場合、ブロックはつぎの例が示すように可能な限り均等に余分な出力要素を分 けます。



ベクトル表現に-1を入力すると、ブロックは自動的に対応する端子に合ったサ イズにします。たとえば、ベクトル表現が[-1,3-1]とすると、2番目の信号は常 に3要素もち、さらに、1番目と2番目の信号のサイズは入力信号のサイズに依 存します。

ベクトル表現が正の値と-1からなる場合、ブロックは正の値が端子に必要とす る要素の数だけ割り当て、残りの要素を-1の端子にできるだけ均等になるよう に分配します。たとえば、ブロック入力が7要素幅で、[-1,3-1]と出力を指定す ると仮定します。この場合、ブロックは1番目の端子に2要素、2番目に3要 素、そして3番目に2要素出力します。



バス選択モードでの出力数の指定

バス選択モードでは、Number of outputs パラメータの値はつぎの値をとることが できます。 • スカラ値で出力端子の数を指定

指定した値は、入力信号の数と一致しなければなりません。たとえば、バス 入力が2信号からなり、このパラメータの値がスカラーの場合、値は2に等 しくしてください。



- ベクトルで対応する端子に出力する信号の数を指定します。
 たとえば、バス入力が5信号からなる場合、出力として[3,2]と指定することができます。この場合、ブロックは1番目の端子に入力信号のうちの3信号を出力し、2番目の端子に残りの2信号を出力します。
- ベクトルのセル配列要素からなるセル配列で、対応する出力によって信号出力の次元を指定します。

セル配列の書式では、特定の次元の信号だけを入力するように Demux ブロック を制限します。たとえば、セル配列 {{[22],3} {1}}は、2 × 2 行列、3 要素ベク トルおよび1 スカラー信号からなるバス信号だけを入力するブロックをあらわし ます。自動的に入力に基づいた出力の次元を決めるには、セル配列式に-1 を使 用します。たとえば、つぎのダイアグラムは、一番左の Demux プロックの出力 を指定するためにセル配列式 {{-1}, {-1,-1}}を使用しています。



バス選択モードでは、出力端子の次元を指定する場合、つまり-1以外の任意の 値を指定する場合、対応する入力要素は指定した次元と一致していなければなり ません。

注意 Simulink は、Simulink ライブラリからモデルへ Demux ブロックをコピーし た場合、Demux ブロックの名前を非表示にします。

Demux ブロックは、任意の数値(複素数または実数)やデータタイプの信号を受 サポートされて いるデータタイ け入れ、出力します。 プ

バラメータとタ	Block Parameters: Demux	×
イアログボック ス	Demux Split bus signals into their constituent scalar, vector, or matrix signals. Parameters Number of outputs:	
	OK Cancel Help Apply	

Number of outputs

出力の数と次元。

Bus selection mode

バス選択モードを利用可能にする。

Derivative

目的 入力の時間微係数を出力します。

ライブラリ Continuous

詳細

Derivative ブロックは、つぎの計算を行うことによって、その入力の微係数を近似します。



 $\frac{\Delta u}{\Delta t}$

ここで、Δ*u* は入力値の変位、Δ*t* は前のシミュレーション時間ステップからの時間の変位です。プロックは、1つの入力を受け入れ、1つの出力を生成します。 シミュレーション開始前の入力信号の値は、ゼロと見なされます。プロックに対する初期出力はゼロです。

結果の精度は、シミュレーションで使われる時間ステップのサイズによって異な ります。ステップが小さいほど、より滑らかで正確な出力曲線がこのプロックか ら得られます。連続状態をもつブロックと異なり、入力が急速に変化するとき に、ソルバはより小さなステップは使いません。

入力が離散信号の場合、入力の連続微係数は、入力の値が変化するときにはイン パルスで、そうでないときには0です。次式を使って、離散信号の離散微係数を 求めることができます。

$$y(k) = \frac{1}{\Delta t}(u(k) - u(k-1))$$

つぎのz変換を行います。

$$\frac{Y(z)}{u(z)} = \frac{1-z^{-1}}{\Delta t} = \frac{z-1}{\Delta t \cdot z}$$

linmod を使った Derivative ブロックを含むモデルの線形化は、問題を含んでいます。この問題をどのように回避するかについては、6-4 ページの"線形化"を参照してください。

Derivative ブロックは、double タイプの実数信号を受け入れ、出力します。

サポートされて いるデータタイ プ

ダイアログボッ クス

Block Parameters: D	erivative			×
Derivative				
Numerical derivative	e: du/dt.			
OK	Cancel	<u>H</u> elp	Apply	

直接フィードスルー	あり
サンプル時間	連続
スカラ拡張	N/A
状態	0
次元化	可
ゼロクロッシング	なし

Digital Clock

目的 指定したサンプリング間隔でシミュレーション時間を出力します。

ライブラリ Sources

詳細 Digital Clock ブロックは、指定したサンプリング間隔でのみ、シミュレーション 時間を出力します。それ以外の時間では、出力は前の値のままです。

12:34 ♪ 離散システム内で現在の時間を必要とする場合、(連続時間を出力する)Clock ブ ロックではなく、このブロックを使用してください。

れて Digital Clock ブロックは、double タイプの実数信号を出力します。

サポートされて いるデータタイ プ

パラ	メーク	レダ
イア	ログオ	、 ック
ス		

Block Parameters: Digital Clock	×
Output current simulation time at the specified rate.	
Parameters	
Sample time:	
1	
OK Cancel <u>H</u> elp <u>A</u> pply	

Sample time

サンプリング間隔。デフォルト値は1秒です。

サンプル時間	離散
スカラ拡張	不可
次元化	不可
ゼロクロッシング	なし
目的 スカラ、ベクトル、2次元行列を取得するためのN次元テーブルのインデックス

ライブラリ Functions & Tables

詳細



Direct Look-Up Table (n-D) ブロックは、ブロック入力を n 次テーブルのゼロベー スのインデックスとして使用します。入力数は、希望する出力の形状により異な ります。出力は、スカラ、ベクトル、2 次元行列です。ルックアップテーブル は、ゼロベースのインデックスを利用するので、整数のデータタイプはこの範囲 全体を示すことができます。たとえば、uint8 データタイプを使ったテーブルの 大きさは、256 要素を表わすことができます。

出力値の組を Table data パラメータとして定義します。出力の形状をスカラ、ベクトル、2次元行列として指定します。つぎの図に示すように、最初の入力は出力の次元数よりも大きい最初の次元のゼロベースのインデックスを指定し、2番目の入力はつぎのテーブルの次元のインデックスを指定、のように指定します。

図は、出力形状が "2-D Matrix" に設定された 5 次元テーブルを示しています。出 力は、R 行 C 列の 2 次元行列です。 つぎの図は、(ブロックのダイアログボックスでどのオプションを選択したかに より)Direct Look-Up Table ブロックが表わすすべてのアイコンを示しています。



4 よりも大きい次元については、アイコンは 4-D アイコンと一致しますが、テキ ストの一番上に "8-D T[k]" のように正確な次元数を表示します。アイコンの一番 上の行は、ブロック出力が 1 つまたは複数要素のテーブルルックアップから得ら れるときに用いられます。"n-D Direct Table Lookup5" とラベル付けられたブロッ ク 6, 8, 12 は、テーブルから列を抽出するように設定され、7 および 9 で終わる 2 つのプロックはテーブルから面を抽出します。10, 11 および 12 で終わる figure の ブロックは、テーブルをパラメータではなく入力として設定します。

例題

つぎの例で、ブロックパラメータは以下のように定義されます。

Invalid input value: "Clip and Warn" Output shape: "Vector" Table data: int16(a)

ここで、aは MATLAB を使って計算される線形増加する数値をもつ4次元配列です。

a = ones(20,4,5,7); L = prod(size(a));a(1:L) = [1:L]'; テーブルのインデックスはゼロベースで、figure は、ブロックが4次元目の3番目の要素から、3次元目の4番目の要素の2列目の20個の値からなるベクトルを出力することを示します。



この例での出力値は、(1 ベースのインデックス付けを用いる)MATLAB でマ ニュアルで計算できます

a(:,1+1,1+3,1+2)

ans =

サポートされて いるデータタイ プ

Direct Look-Up Table (n-D) ブロックは、double, single, int8, uint8, int16, uint16, int32, uint32 タイプの混在する信号を受け入れます。出力タイプは、入力タイプと異なり、入力についてリストされた任意のタイプでかまいません。出力タイプは、 Table data パラメータのデータタイプから継承されます。

テーブルが入力端子においてブロックに接続する場合、出力端子のタイプはテー ブルの入力端子から継承されます。インデックス付けのための入力は、実数でな ければなりません。テーブルのデータは複素数でもかまいません。

ダイアログボッ クス

Block Parameters: Direct Look-Up Table (n-D)
LookupNDDirect (mask) (link)
Table member selection. Inputs are zero-based indices into the table, e.g., an input of 3 returns the fourth element in that dimension. Block can also be used to select a column or 2-D matrix out of the table.
Parameters
Number of table dimensions:
Inputs select this object from table: Element
🥅 Make table an input
Table data:
[4 5 6;16 19 20;10 18 23]
Action for out of range input: Warning
OK Cancel Help Apply

Number of table dimensions(テーブル次元の数)

Table data パラメータは次元の数をもっていなければなりません。この次元の数 によってテーブルの独立変数の数が決まり、プロック入力の数が決まります(詳 細は以下の "Explicit Number of dimensions" および "Use one (vector) input port instead of N ports," を参照してください)。

Inputs select this object from table(入力がテーブルからこのオブジェクトを選択)

出力データがシングル要素か、n次元列行列か、あるいは2次元行列かを指 定します。

Element — # of ports = # of dimensions

Column — # of ports = # of dimensions - 1

2-D Matrix — # of ports = # of dimension -2

この番号付けは、MATLAB の番号付けに従います。たとえば、4 次元の テーブルデータの場合、シングル要素にアクセスするために、配列 (1,2,3,4) のように、4 つのインデックスを指定しなければなりません。行を指定する ためには、配列 (:,2,3,4) のような、3 つのインデックスが必要です。最後に 2 次元行列を指定する場合は、配列 (:,:,3,4) のように、2 つのインデックスが 必要です。

Make table an input(入力テーブルの作成)

このボックスをチェックすると、Direct Look-Up Table (n-D) は Table Data パ ラメータを無視します。代りに、隣に"T"という新しい端子が表示されます。 テーブルデータを入力するときはこの端子を使用します。

Table data $(\mathcal{F} - \mathcal{J} \mathcal{W} \mathcal{F} - \mathcal{P})$

出力値のテーブル。次元の数が4を超えた場合、行列のサイズはN breakpoint set パラメータや Explicit number of dimensions パラメータで定義 した次元に合っていなければなりません。ブロックダイアグラムを修正して いる間は、Table data のフィールドを空のままにしておけますが、シミュ レーションの間は、Table data の次元の数を Number of table dimensions に合わ さなければなりません。MATALB で多次元配列を構築する方法については、 MATALB のオンラインドキュメンテーションの Multidimensional Arrays をご 覧ください。

Action for out of range input(入力範囲外の挙動)

None, Warning, Error.

Real-Time Workshop の注意: 生成コード内で、"Clip and Warn" および "Clip Index" オプションにより、Real-Time Workshop はワーニングを生成するため のコードを含まない clipping コードを生成します。それ以外のオプション "Generate Error" で生成したコードは、clipping コードやエラーメッセージを 含まず、シミュレーションがプロジェクトの設計フェーズ中では範囲外の場 合となるモデルの欠点を明らかにする、という仮定のもとに動作します。この仮定は、生成したコードを非常に効率的にするために役立ちます。

特性 直接フィードスルー あり サンプル時間 接続されるブロックから継承

スカラ拡張	スカラのルックアップについてのみ(テーブルから列 または2次元行列を出力しないとき)
次元化	スカラのルックアップについてのみ(テーブルから列 または2次元行列を出力しないとき)
ゼロクロッシング	なし

目的 IIR フィルタと FIR フィルタを実現します。

ライブラリ Discrete

詳細



Discrete Filter ブロックは、無限インパルス応答 (Infinite Impulse Response (IIR)) と 有限インパルス応答 (Finite Impulse Response (FIR)) を実行します。Numerator と Denominator パラメータを使用した ベクトルとして z^{-1} の次数の低い順に、分子と 分母の多項式の係数を指定します。分母の次数は、分子の次数に等しいかそれよ り大きくなるようにしてください。係数に関する詳細は、9-82 ページの Discrete Transfer Fcn を参照してください。

Discrete Filter ブロックは、信号処理エンジニアがよく利用する方法、つまり、 z^{-l} (遅れ演算子)の多項式を使ってディジタルフィルタを記述します。Discrete Transfer Fcn ブロックは、制御エンジニアがよく利用する方法、つまり、zの多項 式を使って離散システムを記述します、分子多項式と分母多項式を同じ長さて定 義すると、これらの方法は等しくなります。n要素のベクトルは、次数n-1の多 項式を表わします。

ブロックアイコンは、指定した分子と分母を表示します。Simulink がアイコンを どのように表示するかに関しては9-243 ページの "Transfer Fcn" を参照してくださ い。

サポートされて いるデータタイ プ ブロックは、double タイプの実数信号を入出力します。

パラメータとダ イアログボック ス

Block Parameters: Discrete Filter	×
Discrete Filter	
Vector expression for numerator and denominator. Coefficients are for ascending powers of 1/z.	
Parameters	5
Numerator:	
[1]	
Denominator:	
[1 2]	
Sample time:	
1	
OK Cancel <u>H</u> elp <u>A</u> pply	

Numerator

分子係数のベクトル。デフォルトは[1]。

Denominator

分母係数のベクトル。デフォルトは[12]。

Sample time

サンプル間の時間間隔。

特性	直接フィードスルー	Numerator と Denominator パラメータの長さが等しいと きのみ
	サンプル時間	離散
	スカラ拡張	不可
	状態数	Denominator パラメータの長さ -1
	次元化	不可
	ゼロクロッシング	なし

目的 一定間隔でパルスを生成します。

ライブラリ Sources

詳細 Discrete Pulse Generator ブロックは、一定間隔で連続的なパルスを生成します。



つぎのパルスパラメータを指定することができます。パルス幅は、パルスが高い まま続く間をサンプル周期の数で表わしたものです。周期は、パルスが高い部分 と低い部分を合わせたものをサンプル周期で表わしたものです。位相の遅れは、 パルスがスタートする前のサンプル周期の数です。位相の遅れは、正の数または 負の数のいずれをとることもできますが、周期よりも大きい値では設定できませ ん。サンプル時間は、0よりも大きくなければなりません。すべてのパラメータ は任意のスカラーパラメータのスカラー拡張の後で同じ次元にしなければなりま せん。

離散システムやハイブリッド(混合)システムでは Discrete Pulse Generator ブロックを使用してください。連続信号を生成するには、Pulse Generator ブロック (9-175 ページの "Pulse Generator" を参照)を使用してください。

サポートされて Discrete Pulse Generator ブロックは、double タイプの実数信号を受け入れ、出力 **いるデータタイ** します。 プ パラメータとダ イアログボック ス

Block Parameters: Discrete Pulse Generator 🛛 🔀
Discrete Pulse Generator
Generate pulses at regular intervals. Specify parameters as integer multiples of the sample time.
Parameters Amplitude:
1
Period (number of samples):
2
Pulse width (number of samples):
1
Phase delay (number of samples):
0
Sample time:
1
☑ Interpret vector parameters as 1-D
OK Cancel <u>H</u> elp Apply

Amplitude

パルスの振幅で、デフォルトは1。

Period

サンプルの数で示されたパルスの周期。デフォルトは2。

Pulse width

パルスが高いまま続く間を周期の数で表わしたもの。デフォルトは1。

Phase delay

それぞれのパルスが生成される前の遅れをサンプルの数で示したもの。デフォルトは、0。

Sample time

サンプル周期。デフォルトは1。

Interpret vector parameters as 1-D

選択すると、pulse generation パラメータのための列や行マトリクスの値がベクトル出力信号になります。

サンプル時間	離散
スカラ拡張	パラメータによる

特性

次元化 可 ゼロクロッシング なし **目的** 離散状態空間システムを実現します。

ライブラリ Discrete

y(n)=Cx(n)+Du(n) «(n+1)=Ax(n)+Bu(n)

詳細

Discrete State-Space ブロックは、つぎのように記述されるシステムを実現します。 x(n+1) = Ax(n) + Bu(n)y(n) = Cx(n) + Du(n)

ここで、*u* は入力、*x* は状態、*y* は出力です。下の図に示すような行列係数は、つぎの特性をもたなければなりません。

- A は、n 行 n 列の行列でなければなりません。n は状態数です。
- B は、n 行 m 列の行列でなければなりません。m は入力数です。
- Cは、r行n列の行列でなければなりません。rは出力数です。
- Dは、r行m列の行列でなければなりません。



ブロックは、1つの入力を受け入れ、1つの出力を生成します。入力ベクトルの幅は、B行列とD行列の列数で決まります。出力ベクトルの幅は、C行列とD 行列の行数によって決まります。

Simulink は、ゼロを含む行列をスパース行列に変換して、乗算の効率を高めます。

サポートされて Discrete State Space ブロックは、double タイプの実数信号を受け入れ、出力しま いるデータタイ す。

プ

パラメータとダ イアログボック ス

Block Parameters: Discrete State-Space 🛛 🛛 💌
Discrete State Space
Discrete state-space model: x(n+1) = Ax(n) + Bu(n) y(n) = Cx(n) + Du(n)
Parameters
<u>A:</u>
1
B:
1
C:
1
D:
1
Initial conditions:
0
Sample time:
1
UK Lancei <u>H</u> elp <u>A</u> pply

A, B, C, D

上記の方程式で定義されるような行列係数。

Initial conditions

初期状態ベクトル。デフォルトは0。

Sample time

サンプル間の時間間隔。

直接フィードスルー	D≠0の場合のみ
サンプル時間	離散
スカラ拡張	初期条件による
状態数	A のサイズに依存
次元化	可
ゼロクロッシング	なし

目的 信号の離散時間積分を実行します。

ライブラリ Discrete

詳細

 $\left| \frac{T}{z \cdot 1} \right|$

Discrete-Time Integrator ブロックは、純粋に離散システムを構築する場合、 Integrator ブロックの代わりに使用することができます。

Discrete-Time Integrator ブロックを使用すると、以下のことが可能になります。

- 初期条件を、ブロックダイアログボックスで、あるいはブロックに対する入力として定義する
- ブロックの状態を出力する
- 積分に上限と下限を定義する
- 追加のリセット入力に応じて状態をリセットする

これらの特徴について以下に説明します。

積分手法

ブロックは、つぎの手法で積分を行うことができます。Forward Euler(前進型)、 Backward Euler(後退型)、Trapezoidal(中点型)です。Simulink は、与えられたス テップ k に対して、y(k) と x(k+1) を更新します。T はサンプリング周期(トリガ 付きサンプリング時間の場合は delta T)です。値は上限または下限に応じてク リップされます。すべての場合において、x(0)=IC です(必要に応じてクリップ されます)。

• Forward Euler 法(デフォルト)は、Forward Rectangular(前進矩形)または左側近似 としても知られています。

この手法では、1/s は T/(z-1) で近似されます。これによってつぎの式が選られます。

y(k) = y(k-1) + T * u(k-1)

x(k+1) = x(k) + T*u(k) このとき、

- ステップ 0: y(0) = x(0) = IC (必要ならばクリップ) $x(1) = y(0) + T^*u(0)$
- ステップ 1: y(1) = x(1) $x(2) = x(1) + T^*u(1)$

ステップ k: y(k) = x(k)

x(k+1) = x(k) + T*u(k) (必要ならばクリップ)

この手法では、入力端子1は、直接フィードスルーをもちません。

• Backward Euler 法は、Backward Rectangular(後退矩形)または右側近似としても 知られています。

この手法では、1/s は T*z/(z-1) で近似されます。これによってつぎの式が得られます。

y(k) = y(k-1) + T * u(k).

x(k) = y(k-1), このとき、 ステップ 0: y(0) = x(0) = IC (必要ならばクリップ) x(1) = y(0)

ステップ 1:
$$y(1) = x(1) + T^*u(1)$$

 $x(2) = y(1)$

ステップ k: $y(k) = x(k) + T^*u(k)$ x(k+1) = y(k)

この手法では、入力端子1が直接フィードスルーをもちます。

• Trapezoidal 法。この手法の場合、1/s は T/2*(z+1)/(z-1) で近似されます。

T が (サンプリング周期に等しく) 固定されている場合、つぎのようになりま す。 x(k) = y(k-1) + T/2 * u(k-1).

このとき、

ステップ 0: y(0) = x(0) = IC (必要ならばクリップ)x(1) = y(0) + T/2 * u(0)

- ステップ 1: y(1) = x(1) + T/2 * u(1)x(2) = y(1) + T/2 * u(1)
- ステップ k: y(k) = x(k) + T/2 * u(k)x(k+1) = y(k) + T/2 * u(k)

ここで、x(k+1) は次回の出力の最良推定値です。それは x(k) != y(k) という意味 で、状態ではありません。

Tが変数の場合(すなわち、トリガ時間から得られた場合)には、つぎのようになります。

- ステップ 0: y(0) = x(0) = IC (必要ならばクリップ)x(1) = y(0)
- ステップ 1: x(1) = x(1) + T/2 * (u(1) + u(0))x(2) = y(1)
- ステップ k: y(k) = x(k) + T/2 * (u(k) + u(k-1))x(k+1) = y(k)

この手法では、入力端子1が直接フィードスルーをもちます。 ブロックアイコンは、下の図に示すように、選択した積分手法を反映します。



初期条件の定義

初期条件は、ブロックダイアログボックスのパラメータとして、あるいは外部信 号から入力することができます。

 初期条件をブロックパラメータとして定義するためには、Initial condition source パラメータを internal と指定し、Initial condition パラメータフィールド に値を入力します。 外部入力源から初期条件を提供するには、Initial condition source パラメータを external と指定します。下の図に示すように、追加の入力端子がブロック入力 の下に表示されます。



状態端子の利用

既に知られているつぎの2つの状況において、ユーザは、出力端子の代わりに状 態端子を利用しなければなりません。

- ブロックの出力がブロックリセット端子または初期条件端子によってフィードバックされ、代数ループを発生する場合。このような状況の例については、bounceモデルを参照してください。
- 条件付きで実行されるサブシステムから別のサブシステムに状態を受け渡したい場合。この場合には、タイミングの問題が生じます。このような状況の例については、clutchモデルを参照してください。

ユーザは、出力端子ではなく、状態端子を通じて状態の受け渡しを行うことにより、以上のような問題を回避することができます。たとえ値が同一であっても、 Simulink はそれらの数値をやや異なる時間で生成するため、これらの問題を防ぐ ことができます。ブロックの状態は、Show state port チェックボックスを選択す ることによって出力されます。

デフォルトでは、つぎに示すように、状態端子はブロックの上部に表示されます。



積分の制限

出力が指定可能なレベルを越えないように、Limit output チェックボックスを選 択し、限界値を該当するパラメータフィールドに入力します。そうすると、ブ ロックは制限付き積分器と同じ機能をもちます。出力が限界値に近づくと、積分 動作はオフとなり、積分が広がるのを防ぎます。シミュレーション中に限界値を 変更することはできますが、出力が制限付きかそうでないかの状態は変更できま せん。出力は、つぎのようにして決められます。

- 積分値が Lower saturation limit 未満で、入力が負の場合、出力は Lower saturation limit に保持されます。
- 積分値が Lower saturation limit と Upper saturation limit の間にある場合、出力 は積分値となります。
- 積分値が Upper saturation limit より大きくて、入力が正の場合、出力は Upper saturation limit に保持されます。

状態がいつ制限を受けるかを示す信号を生成するには、Show saturation port チェックプロックを選択します。下の図のように、飽和端子がブロック出力端子 の下に表示されます。



信号は、つぎの3つの値のいずれかです。

- 1は、上限が適用されたことを示します。
- 0は、積分が制限されないことを示します。
- -1は、下限が適用されたことを示します。

Limit output オプションを選択すると、ブロックには3つのゼロクロッシングが存在します。1つは飽和上限になるときを検出するもの、1つは飽和下限になるときを検出するもの、そして、もう1つは飽和状態でなくなるときを検出するものです。

状態のリセット

ブロックは、外部信号に基づいて、指定した初期条件にその状態をリセットする ことができます。ブロックに状態をリセットさせるには、External reset の選択 肢の1つを選びます。つぎの図に示すように、ブロックの入力端子の下にトリガ 端子が表示されてトリガタイプを示します。



- rising を選択すると、トリガ信号が正の方向でゼロを横切る瞬間をトリガイベントとします。
- falling を選択すると、トリガ信号が負の方向でゼロを横切る瞬間をトリガイベントとします。
- either を選択すると、トリガ信号が正または負いずれかの方向でゼロを横切る 瞬間をトリガイベントとします。
- level を選択すると、リセット信号が非ゼロの間を トリガイベントとし、出力 値を初期状態にします。

リセット端子には直接フィードスルーがあります。ブロック出力がこの端子に直接、または直接フィードスルーをもつ一連のブロックを通してフィードバックされると、代数ループが発生します。このループを解除するには、代わりにブロック状態をリセット端子にフィードします。ブロックの状態にアクセスするには、Show state port チェックボックスを選択します。

すべてのオプションの選択

すべてのオプションを選択すると、アイコンは、つぎのように表示されます。



サポートされて いるデータタイ プ Discrete-Time Integrator ブロックは、double タイプの実数信号を受け入れ、出力します。

パラメータとダ イアログボック ス

Block Parameters: Discrete-Time Integrator	×
Discrete-Time Integrator	
Discrete-time integration of the input signal.	
Parameters	
Integrator method ForwardEuler	
External reset none	
Initial condition source internal	
Initial condition:	
0	
✓ Limit output	
Upper saturation limit:	
inf	
Lower saturation limit:	
-inf	
Show saturation port	
Show state port	
Sample time:	
1	
OK Cancel <u>H</u> elp Apply	

Integrator method

積分手法。デフォルトは ForwardEuler。

External reset

リセット信号でトリガイベント (rising, falling, either) が生じた場合、状態を初期条件にリセットします。

Initial condition source

Initial condition パラメータ (internal に設定されている場合) または external ブロック (external に設定されている場合) から状態の初期条件を取り出しま す。

Initial condition

状態の初期条件。Initial condition source パラメータを internal に設定したときに有効です。

Limit output

チェックすると、Lower saturation limit パラメータと Upper saturation limit パラメータの間の値に状態を制限します。

Upper saturation limit

積分の上限。デフォルトは inf。

Lower saturation limit

積分の下限。デフォルトは-inf。

Show saturation port

チェックすると、ブロックに飽和出力端子が追加されます。

Show state port

チェックすると、ブロック状態に対するブロックに出力端子が追加されま す。

Sample time

サンプル時間間隔。デフォルトは1。

直接フィードスルーあり、リセットと外部初期条件入力源端子についてサンプル時間離散スカラ拡張パラメータによる状態数接続されるブロックとパラメータから継承次元化可ゼロクロッシングリセット検出のために1つ、飽和上限検出のためと飽
和下限検出のためにそれぞれ1つずつ、そして飽和で
なくなる場合に1つ。

特性

目的 離散伝達関数を実現します。

ライブラリ Discrete

詳細

Discrete Transfer Fcn ブロックは、つぎの方程式で記述される *z* 変換伝達関数を実現します。



$$H(z) = \frac{num(z)}{den(z)} = \frac{num_0 z^n + num_1 z^{n-1} + \dots + num_m z^{n-m}}{den_0 z^n + den_1 z^{n-1} + \dots + den_n}$$

ここで、 $m+1 \ge n+1$ は、それぞれ分子係数と分母係数の数です。 $num \ge den$ は、 z のべき乗の降順での分子と分母の係数を含みます。numはベクトルまたは行列 で、denはベクトルでなければなりません。いずれもブロックダイアログボック ス上のパラメータとして指定します。分母の次数は分子の次数以上でなければな りません。

ブロック入力はスカラです。出力幅は分子の行数に等しくなります。

Discrete Transfer Fcn ブロックは、制御エンジニアがよく利用する方法、つまり、 z の多項式を使って離散システムを記述します。Discrete Filter ブロックは、信号 処理エンジニアがよく利用する方法、つまり、z-1(遅れ演算子)の多項式を使っ てデジタルフィルタを記述します。分子多項式と分母多項式を同じ長さで定義す ると、これらの方法は等しくなります。

Discrete Transfer Fcn ブロックは、分子と分母をどのように指定するかに応じて、 アイコン内に表示します。詳細については、9-255 ページの"Transfer Fcn"を参照 してください。

サポートされて Discrete Transfer Function ブロックは、double タイプの実数信号を受け入れ、出力 いるデータタイ します。

プ

パラメータとダ イアログボック ス

Block Parameters: Discrete Transfer Fcn
Discrete Transfer Fon
Matrix expression for numerator, vector expression for denominator. Dutput width equals the number of rows in the numerator. Coefficients are for descending powers of z.
Parameters
Numerator:
[1]
Denominator:
[1 0.5]
Sample time:
1
OK Cancel <u>H</u> elp Apply

Numerator

分子係数の行ベクトル。複数行の行列を指定すると複数の出力を生成することができます。デフォルトは[1]。

Denominator

分母係数の行ベクトル。デフォルトは[10.5]。

Sample time

サンプル時間間隔。デフォルトは1。

 直接フィードスルー
 Numerator と Denominator のパラメータの長さが等しい場合のみ

 サンプル時間
 離散

 スカラ拡張
 不可

 状態数
 Denominator パラメータの長さ -1

 次元化
 不可

 ゼロクロッシング
 なし

特性

目的 極 - 零点型で指定した離散伝達関数を実現します。

ライブラリ Discrete

詳細



サポートされて

いるデータタイ

プ

Discrete Zero-Pole ブロックは、遅れ演算子 z を使って、零点、極、ゲインをもつ 離散システムを実現します。伝達関数は、因数分解または零点 - 極 - ゲイン型で 表現することができ、MATLABの単入力単出力システムの場合、つぎのように なります。

$$H(z) = K \frac{Z(z)}{P(z)} = K \frac{(z - Z_1)(z - Z_2)...(z - Z_m)}{(z - P_1)(z - P_2)...(z - P_n)}$$

ここで、Zは零点ベクトル、Pは極ベクトル、Kはゲインを表します。極の数は、 零点の数以上でなければなりません ($n \ge m$)。極および零点が複素数の場合、それ らは複素共役対でなければなりません。

このブロックのアイコンは、パラメータをどのように指定するかに応じて伝達関数を表示します。詳細は、9-276ページの "Zero-Pole" を参照してください。

Discrete Zero-Pole ブロックは、double タイプの実数信号を受け入れ、出力しま す。

パラメータとダ イアログボック ス	Block Param Discrete Z Matrix expre width equal vector.
	Parameters
	Zeros:
	[1]

NOCK Parameters, Discrete Zero-Pole	
Discrete Zero-Pole	
Matrix expression for zeros. Vector expression for poles and gain. Output width equals the number of columns in zeros matrix, or one if zeros is a vector.	
Parameters	
Zeros:	
[1]	
Poles:	
[0 0.5]	
Gain:	
1	
Sample time:	
1	
OK Cancel <u>H</u> elp <u>A</u> pply	

Zeros

零点の行列。デフォルトは[1]。

 Poles
 極のベクトル。デフォルトは [0 0.5]。

 Gain
 ゲイン。デフォルトは 1。

 Sample time
 サンプル時間間隔。

 直接フィードスルー
 あり、零点と極の数が等しい場合

 サンプル時間
 離散

 スカラ拡張
 なし

 状態数
 Poles ベクトルの長さ

不可

なし

次元化

ゼロクロッシング

特性

目的 入力の値を示します。

ライブラリ Sinks

詳細

Display ブロックは、入力の値を示します。



表示書式は、Format 選択肢を選択することによって制御することができます。

- short は、5桁のスケーリングされた固定小数点値を表示します。
- long は、15桁のスケーリングされた固定小数点値を表示します。
- short_eは、5桁の浮動小数点値を表示します。
- long_eは、16桁の浮動小数点値を表示します。
- bank は、固定したドルとセントの書式で値を表示します(ただし、\$やコンマは 付きません)。

フローティング表示で Display ブロックを利用するには、Floating display チェッ クボックスを選択します。このチェックボックスを選択すると、Display ブロッ クの入力端子は表示されなくなり、選択されたライン上の信号値を表示します。 Floating display オプションを選択した場合、Simulinkの信号再生機能をオフにし ておかなければなりません。詳細は、5-31 ページの"信号ストレージの利用"を 参照してください。

表示されるデータの量とデータを表示する時間ステップは、ブロックパラメータ で決まります。

- Decimation パラメータでは、n番目のサンプルごとにデータを表示することができます。ここで、nは間引きファクタです。デフォルトの間引きファクタは1で、各時間ステップごとにデータを表示します。
- Sample time パラメータでは、ポイントを表示するサンプリング間隔を指定することができます。このパラメータは、時間ステップ間隔が同じでない可変ステップソルバを使用する場合に役立ちます。デフォルト値は-1 で、どのポイントを表示するかを決めるときに、ブロックはサンプリング間隔を無視します。

ブロック入力がベクトルの場合、最初の要素だけでなくその他の要素も表示する ようにブロックのサイズを変更することができます。ブロックの大きさは垂直、 水平いずれかの方向で変更することができます。ブロックは、適切な方向で表示 フィールドを追加します。黒の三角形は、ブロックがすべての入力ベクトル要素 を表示していないことを示します。たとえば、つぎの図は、Display ブロックへ の入力がベクトルであるモデルを示しています。上のモデルはサイズを変更する

前のブロックを示しています。黒の三角形に注意してください。下のモデルは2 つの入力要素を表示するようにサイズを変更したブロックを示しています。



サポートされて いるデータタイ プ

パラメータとダ イアログボック ス Display ブロックは、任意のデータタイプの実数または複素数信号を受け入れ、 出力します。

Block Param	eters: Display	×
Display —		
Numeric dis	play of input values.	
- Parameter:	\$	
Format:	short 💌	
Decimati	ion:	
1		
Float	ing display	
Sample 1	Fime:	
-1		
0	IK Cancel <u>H</u> elp <u>Apply</u>	

Format

表示するデータの書式。デフォルトは short。

Decimation

データの表示頻度。デフォルトは1で、すべての入力点を表示。

Floating display

チェックすると、ブロックの入力端子は表示されなくなり、ブロックをフ ローティング Display ブロックとして使用することができます。

Display

Sample time

表示を更新する時間。

特性

サンプル時間 接続されるブロックから継承 次元化 可 目的 内積を計算します。

ライブラリ Math

詳細

Dot Product ブロックは、2つの入力ベクトルの内積を計算します。スカラ出力 y は、つぎの MATLAB 演算と等価です。

y = sum(conj(u1) .* u2)

ここで、u1とu2はベクトル入力を示しています。2つの入力がベクトルである 場合、それらは互いに同じ長さでなくてはなりません。入力ベクトルの要素は、 double タイプの実数値または複素数信号値です。出力信号のタイプ(複素信号か 実数信号)は、入力信号のタイプに依存します。

入力1	入力 2	出力
実数	実数	実数
実数	複素数	複素数
複素数	実数	複素数
複素数	複素数	複素数

要素同士を加算することなく乗算する場合は、Product ブロックを使用してくだ さい。

Dot Product ブロックは、double タイプの信号を受け入れ、出力します。

サポートされて いるデータタイ プ

ダイアログボッ	Block Parameters: Dot Product	×
	Dot Product (mask) (link)	
9X	Inner (dot) product. y = sum[conj(u1).*u2]	
	OK Cancel Help Apply	

特性

直接フィードスルー あり

> サンプル時間 接続されるブロックから継承

Dot Product

スカラ拡張	不可
状態数	0
次元化	可
ゼロクロッシング	なし

目的 Enable ポートをサブシステムに追加します。

ライブラリ Signals & Systems

詳細 Enable ブロックをサブシステムに追加すると、Enabled サブシステムとなりま す。Enabled サブシステムは、イネーブル端子で受け取る入力が、ゼロより大き い間実行されます。

> シミュレーションの開始時に、Simulink は、Enabled サブシステム内にあるブ ロックの状態を初期条件により初期化します。Enabled サブシステムの再起動時 (使用不可能となっていた後の実行時)に、States when enabling パラメータは、 Enabled サブシステムを構成するブロックの場状態がどのようになるかを決定し ます。

- reset は、状態を初期条件(定義されていない場合はゼロ)にリセットします。
- held は、状態を前の値に保持します。

イネーブル信号は、Show output port チェックボックスを選択することによって 出力することができます。このオプションを選択すると、システムはイネーブル 信号を処理することができます。信号の幅は、イネーブル信号の幅です。

サブシステムは、複数の Enable ブロックを含むことはできません。

サポートされて Enable ブロックの入力端子のデータタイプは、任意のタイプにすることができ いるデータタイ ます。Enabled サブシステムに関する詳細は第8章の"条件付きで実行されるサブ プ システム"を参照してください。

パラメータとダ イマログボック	Block Parameters: Enable Enable Port	×	
イアロクホックス	Place this block in a subsystem to create an enabled subsystem. Parameters States when enabling: held		
	C Show output port K Cancel Help Apply		

States when enabling

サブシステムが再び Enabled になったとき、内部の状態量をどのように扱う かを指定します。

Show output port

チェックすると、Simulink は Enable ブロックに出力端子を表示し、Enable 信号を出力します。

特性	サンプル時間	Enable 端子の信号によって決まります。
	次元化	可

目的 指定した式を入力に適用します。

ライブラリ Functions & Tables

詳細

f(u)

- Fcn ブロックは、指定した C 言語スタイル表現をその入力に適用します。式は、 つぎの成分から構成することができます。
- u ブロックへの入力。u がベクトルの場合、u(i) はベクトルのi番目の要素を 表わします。u(1) はまたは u 単体は最初の要素を表わします。
- 数値定数
- 算術演算子 (+-*/)
- 比較演算子 (== !=><>= <=) 関係が TRUE(真) であれば式は1を出力し、そうでなければ0を出力します。
- ・論理演算子(&&||!) ― 関係がTRUE(真)であれば式は1を出力し、そうでなければ0を出力します。
- 括弧
- 数学関係 abs, acos, asin, atan, atan2, ceil, cos, cosh, exp, fabs, floor, hypot, ln, log, log10, pow, power, rem, sgn, sin, sinh, sqrt, tan, and tanh.
- ワークスペース変数 上の項目リストにない変数名は、MATLAB に渡されて 評価されます。行列またはベクトル要素は、具体的に参照する必要がありま す(たとえば、行列の最初の要素の場合は A ではなく A(1,1))。

優先順位の規則は、C 言語の基準に従います。

- 1 ()
- 2 +-(単項)
- 3 pow(累乗)
- 4 !
- 5 */
- 6 + -
- 7 > < <= >=
- **8** = !=

9 &&

10 ||

式は、行列計算を実行できないところが MATLAB 式と異なります。また、この ブロックは、コロン演算子 (:) をサポートしていません。

ブロック入力はスカラでもベクトルでも構いません。出力は常にスカラです。ベクトル出力の場合、Math Function ブロックを使用することを考えてください。 ブロック入力がベクトルで、関数が入力要素に対して個々に演算を行う場合(た とえば、関数 sin)、ブロックは最初のベクトル要素に対してのみ演算を行います。

サポートされて Fcn ブロックは、double タイプの信号を受け入れ、出力します。 いるデータタイ

いるテータタイ プ パニュータトダ

77	フ	ァ	_	7	C	2
1	ア	П	グ	ボ	ッ	ク
ス						

Block Parameters: Fon	×
Fon	
General expression block. Use "u" as the input variable name. Example: sin(u[1] * exp(2:3 * -u[2]))	
Parameters	5
Expression:	
sin(u[1]*exp(2.3*(-u[2])))	
OK Cancel <u>H</u> elp <u>Apply</u>	

Expression

. .

_. . .

入力に適用される C 言語スタイル表現。式の成分は前のページにリストした ものです。式は、数学的に正しく指定する必要があります(すなわち、括弧 の一致、適切な関数引数の数など)。

特性

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	不可
次元化	不可
ゼロクロッシング	なし

目的 1次サンプルアンドホールドを実現します。

ライブラリ Discrete

詳細



First-Order Hold ブロックは、指定したサンプリング間隔で、1次サンプルアンドホールドを実現します。このブロックは実際のアプリケーションではほとんど価値がなく、主に学術目的のために含まれています。

デモプログラム fohdemo を実行すると、Zero-Order Hold ブロックと First-Order Hold ブロックの違いを見ることができます。つぎの図は、Sine Wave ブロックか らの出力と First-Order Hold ブロックからの出力を比較したものです。



サポートされて First-Order Hold ブロックは、double タイプの信号を受け入れ、出力します。 いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: First-Order Hold	×
First Order Hold (mask) (link)	
First Order Hold	
Parameters	
Sample time:	
1	
OK Cancel <u>H</u> elp <u>A</u> pply	

Sample time

サンプル間の時間間隔

特性

直接フィードスルー	なし
サンプル時間	連続
スカラ拡張	不可
状態	入力要素毎に1つの連続状態と1つの離散状態

次元化 可 ゼロクロッシング なし
目的 Goto ブロックからの入力を受け入れます。

ライブラリ Signals & Systems

詳細

 $|A\rangle$

From ブロックは、対応する Goto ブロックからの信号を受け入れ、それを出力とします。出力のデータタイプは、Goto ブロックからの入力のデータタイプと同じものです。From ブロックと Goto ブロックを使用すると、それらを実際に接続しなくても、あるブロックから別のブロックへ信号を渡すことができます。Goto ブロックを From ブロックに関連付けるには、Goto tag パラメータに Goto ブロックのタグを入力します。

Goto ブロックはその信号を複数の From ブロックに渡すことができますが、 From ブロックは1つの Goto ブロックからしか信号を受け取ることができません。

つぎの図は、Goto ブロックと From ブロックを使用することは、それらのブロッ クに接続されているブロック同士を接続するのと等価であることを示していま す。左側のモデルで、Block1 は信号を Block2 に渡します。このモデルは、右側 のモデルと等価で、右側のモデルは Block1 を Goto ブロックに接続し、その信号 を From ブロックに渡した後で Block2 に渡します。



関連付けた Goto ブロックと From ブロックは、つぎの例外を除いてモデルの任 意の位置に表示することができます。どちらかのブロックが条件付きで実行され るサブシステム内にある場合、もう一方のブロックも同じサブシステム内か、(条件付きで実行される別のサブシステムではない)モデル階層下のサブシステム 内になければなりません。しかし、Goto ブロックが状態端子に接続される場合、 信号を条件付きで実行される別のサブシステムの詳細については、第7章を参照してく ださい。

Goto ブロックのタグの表示によって、その信号を受け取ることができる From ブロックが決まります。詳細は、9-111 ページの Goto と 9-114 ページの Goto Tag Visibility を参照してください。ブロックアイコンは、Goto ブロックのタグの表示 を示しています。

- ・ ローカルタグ名は、角括弧 ([]) で囲みます。
- ・範囲を限定したタグ名は中括弧({})で囲みます。
- グローバルタグ名は、追加のキャラクタなしで表示します。

From ブロックは、任意の実数または複素数データタイプの信号を出力します。 サポートされて いるデータタイ プ バラメータとダ Block Parameters: From × イアログボック From Receive signals from the Goto block with the specified tag. If the tag is defined as 'scoped' in the Goto block, then a Goto Tag Visibility block must be used to define the visibility of the tag. After 'Update Diagram', the block icon displays the selected tag name (local tags are enclosed in brackets, ス [], and scoped tag names are enclosed in braces, {}). Parameters Goto tag: A

<u>H</u>elp

Goto tag

この From ブロックに信号を渡す Goto ブロックのタグ

Apply

特性

サンプル時間 次元化

ΟK

Goro ブロックに接続されるブロックから継承 可

Cancel

目的ファイルからデータを読み込みます。

ライブラリ Sources

詳細

From File ブロックは、指定したファイルから読み込んだデータを出力します。 ブロックアイコンはデータを提供するファイル名を表示します。

untitled.mat

ファイルには2行以上の行列が含まれていなければなりません。最初の行には単 調増加する時間列が含まれていなければなりません。その他の行には、各時間に 対応するデータ点が列単位で含まれています。行列は、つぎの形式をもちます。

 $\begin{array}{ccccc} t_1 & t_2 & \dots t_{final} \\ u 1_1 & u 1_2 & \dots u 1_{final} \\ \dots & & \\ u n_1 & u n_2 & \dots u n_{final} \end{array}$

出力の幅は、ファイル内の行数に依存します。ブロックは時間データを使って、 その出力を決定しますが、時間を出力しません。すなわち、m行の行列において は、該当する列の最初の行以外のすべての行のデータから構成される長さが m-1 のベクトルが、ブロックから出力されます。

ファイル内の2つの値の間の時間における出力値が必要な場合、該当する値の間 で線形補間が行われます。必要な時間がファイル内の最初の時間値より小さい場 合や、最後の時間値より大きい場合、Simulink は最初の2つまたは最後の2つの 点を用いて外挿し、値を計算します。

同じ時間値に対して複数の列が行列に含まれる場合、出力は、最初に出会う列に 対するデータ点になります。たとえば、つぎのデータをもつ行列

time values: 0122 data points: 2345

に対して、時刻2で、出力は、この時刻で出会う最初の列に対するデータ点4になります。

Simulink は、シミュレーションの開始時にファイルをメモリに読み込みます。その結果、同じモデル内の To File ブロックで同じ名前のファイルを指定していても、そのファイルからデータを読み込むことはできません。

To File または To Workspace ブロックで保存したデータの使用 From File ブロックは、To File ブロックで書き出されたデータを、修正しないで 読み込むことができます。To Workspace ブロックで書き出されたデータや、 ファイルに保存されたデータを読み込むためには、

- データにはシミュレーション時間が含まれていなければなりません。シミュレーション出力に時間データを含める最も簡単な方法は、Simulation Parameters ダイアログボックスの Workspace I/O ページの時間に変数を指定することです。詳細については、第4章の"モデルの作成"を参照してください。
- ワークスペースに書き出されるデータの形式は、From File ブロックが期待する 形式とは異なります。データをファイルに保存する前に、それを転置してく ださい。From File ブロックで読み込むときに、データは正しい形式になりま す。
- サポートされて From File ブロックは、double タイプの実数信号を出力します。 いるデータタイ

プ パラメータとダ イアログボック

ス

- From File				
Read time and o	utput values from	the first matrix i	n the specified M/	AT file.
The matrix must of correspond to out	contain time value itput elements. Ir	es in row one. A iterpolates betw	een columns.	
Parameters				
File name:				
untitled.mat				
Sample time:				
0				

File name

入力として使用するデータを含むファイルのパス名またはファイル名。デ フォルトのファイル名は untitled.mat です。ファイル名を指定する場合、 Simulink はファイルが MATLAB の作業ディレクトリにあると仮定します(作業ディレクトリを決定するには、MATLAB プロンプトで pwd とタイプし てください)。Simulink が作業ディレクトリで指定したファイル名を見つけ られないときは、エラーメッセージを表示します。

Sample time

ファイルから読み込むデータのサンプルレート

特性

接続されるブロックから継承
不可
1 次元のみ
なし

目的 ワークスペースからデータを読み込みます。

ライブラリ Sources

詳細

simin

From Workspace ブロックは、MATLAB ワークスペースからデータを読み込みま す。ブロックの Data パラメータは、信号値や時間ステップの表を含む行列また は構造体を計算する MATLAB 表現を通してワークスペースのデータを指定しま す。行列または構造体の書式は、ワークスペースからデータをロードするときに 使われるものと同じです (5-19 ページの"ベースワークスペースから入力の読み 込み"を参照)。From Workspace アイコンは、Data パラメータの中の式を表わし ています。

注意 ワークスペースから行列(2次元)データを読み込むためには、 structure-with-time フォーマットを使用しなければなりません。スカラーあるいは ベクトル(1次元)データの読み込みには、配列あるいは構造体フォーマットが 使用できます。

From Workspace ブロックの Interpolate data パラメータは、ワークスペースデー タが与えられている時間の間の時間でのブロック出力を決めます。この Interpolate data オプションを選択すると、ワークスペースのデータで与えられて いる時間と時間の間のステップのデータ値を補間して与えます。このオプション を使わない場合、データが存在する最も近い時刻の値が使用されます。

ブロックの Form output after final data value by パラメータは、最終時間ステップ 後にワークスペースデータをどのように利用して出力するかを決めます。つぎの 表は、パラメータオプションに基づいて、ブロック出力をまとめたものです。

出力オプション の形式	Interpolate オ プション	最終データ後のプロック出力
Extrapolate	On	最終データ値から外挿
Extrapolate	Off	エラー
SettingToZero	On	ゼロ
SettingToZero	Off	ゼロ
HoldingFinalValue	On	最終データ値

出力オプション の形式	Interpolate オ プション	最終データ後のプロック出力
HoldingFinalValue	Off	最終データ値
CyclicRepetition	On	エラー
CyclicRepetition	Off	データの繰り返し。このオプション は、時間を伴わない構造体フォー マットのデータでのみ有効です。

入力表の中に、同じ時間ステップで複数の要素が含まれる場合、Simulink は最終 の要素で指定された信号を使用します。たとえば、つぎのデータを含む入力表を 考えます。

time: 0122 signal: 2345

時刻2で、最新のデータ点は5なので、出力は5になります。

注意 From Workspace ブロックは、出力の書式が構造体または時間を含む構造体 である場合は、To Workspace ブロック(詳細は、9-240 ページの"To Workspace" を参照)の出力を直接読み込むことができます(詳細は、9-240 ページの"To Workspace"を参照)。To Workspace ブロックによって書き出された行列を読み込 むためには、行列に時間の列を加える必要があります。

サポートされる データタイプ データタイプ ドrom Workspace ブロックは、ワークスペースからの任意のタイプの実数および 複素数信号を受け取ります。タイプ double の実数信号は、構造体または行列書 式でかまいません。double 以外の任意のタイプの複素数信号および実数信号は、 構造体書式でなければなりません。

パラメータとダ イアログボック ス

Block Parameters: From Workspace
From Workspace
Read data values specified in array or structure format from MATLAB's workspace. Array (or matix) format: 1-D signal: var=[TimeValues DataValues] For 2-D signal use structure format Structure format: var.time=[TimeValues] var.signals.values=[DataValues] var.signals.dimensions=[DimValues] Select interpolation to interpolate or extrapolate at time steps for which data does not exist.
Parameters Data:
simin
Sample time:
0
Interpolate data
Form output after final data value by: Extrapolation
OK Cancel <u>H</u> elp Apply

Data

シミュレーション時間とそれに対応する信号値からなる表を含んだ行列また は構造体を表わす表現。たとえば、Tと名付けた時間の列ベクトルとUと名 付けた時間に対応する信号値行列をもつワークスペースを考えましょう。こ のパラメータに対するデフォルトの表現 [T,U] は、必要な入力表を含む行列 を与えます。必要な時刻とそれに対応する信号行列または構造体がワークス ペース内に存在する場合、このフィールドに構造体または行列の名前を単に 入力してください。

Sample time

ワークスペースからのデータのサンプルレート

Interpolate data

このオプションは、ブロックに対応する時刻のデータが存在しない場合、線 形に内挿されます。このオプションを使わない場合、カレントの出力は、 データが存在する最も近い時刻での出力になります。

Form output after final data value by

最終時間が過ぎた後、どのようにしてワークスペースの値を利用して出力す るかを選択します。 特性

サンプル時間	接続されるブロックから継承
スカラ拡張	不可
次元化	可
ゼロクロッシング	なし

目的 指定した時間間隔で、Function-Call サブシステムを実行します。

ライブラリ Signals & Systems

詳細



Function-Call Generator ブロックは、このブロックの Sample time パラメータで設 定された時間間隔で、function-call サブシステム (たとえば、function-call システム として構成された Stateflow の状態チャート)を実行します。複数の function-call サブシステムを指定した順序で実行するためには、まず、Function-Call Generator ブロックをコントロールする function-call サブシステムと同じ数の出力端子をも つ Demux ブロックに接続します。その後で、コントロールされるシステムと Demux ブロックの出力とを結線します。このとき、まず、最初の Demux 端子と 結線したシステムが実行され、そのつぎに、2番目の Demux 端子と結線したシ ステム、等々の順に実行されます。

サポートされて いるデータタイ プ

Function-Call ブロックは、double タイプの実数信号を出力します。

パラメータレダ	
イフロゲザック	Function-Call Generato
1 ゲロク <i>かック</i>	Execute a function-call number of times.
ス	Demux the block's outp a prescribed order. The executed first, the syste second, and so on.
	Parameters Sample time:

Call Genera (mask) (link)

Sample time

サンプル間の時間間隔

Number of iterations

時間ステップあたりのブロック実行回数

直接フィードスルー	なし
サンプル時間	ユーザ指定
スカラ拡張	不可

特性

次元化 可 ゼロクロッシング なし **目的** ブロック入力に乗算を実行します。

ライブラリ Math

詳細



Gain ブロックは、指定したゲインファクタを入力と乗算して出力を生成します。 ゲインは、数値または Gain パラメータフィールドの変数または式として入力す ることができます。入力とゲインは、スカラ、ベクトル、行列でもかまいませ ん。Multiplication パラメータを使って要素毎の乗算か、または入力とゲインの 行列乗算を用いるかを指定します。

Gain ブロックアイコンは、ブロックが十分なサイズであれば、Gain パラメータ フィールドに入力した値を表示します。ゲインを変数として指定すると、ブロッ クは変数名を表示します。

スライダコントロールを使ってシミュレーション中にゲインの変更を行うために は、9-221 ページの "Slider Gain" を参照してください。

サポートされる Gain ブロックのデータタイプに対するサポートは、行列乗算か、または要素毎 **データタイプ** の乗算を選択したかにより異なります。

> 行列乗算に対して、入力とゲインは、タイプ single または double の実数値または 複素数値でなければなりません。

> 要素毎の乗算に対しては、Gain ブロックは boolean 以外の任意のタイプの実数お よび複素数のスカラ、ベクトル、行列入力を受け取り、入力と同じデータタイプ の信号を出力します。入力ベクトルの要素は、同じタイプでなければなりませ ん。Gain ブロックの Gain パラメータは、任意のデータタイプの実数または複素 数値のスカラ、ベクトル、行列でかまいません。Gain ブロックは、つぎのタイ プの規則に従います。

- 入力が実数で、ゲインが複素数の場合、出力は複素数になります。
- Gain パラメータのデータタイプが、入力信号のデータタイプと異なり、入力の データタイプがゲインを表わすことができる場合、Simulink は、出力を計算す る前にゲインを入力データタイプに変換します。入力のデータタイプでゲイ ンを表わすことができない場合、Simulink はシミュレーションを停止し、エ ラーを出力します。たとえば、入力データタイプが uint8 で、ゲインが-1 の場 合、エラーになります。ゲインパラメータを精度を落して、入力データタイ プと同じタイプにすることができる場合、Simulink はワーニングを表示しなが ら、シミュレーションを続けます。
- 出力データタイプが整数で、Gain ブロックの Saturate on integer overflow オプ ションが選択される場合、出力が、ブロックの出力データタイプで表わされ

る最大値を超える場合にブロックは飽和します。言い換えれば、出力データ タイプによって表現される最大正数または最小負数が出力されます。たとえ ば、出力タイプが int8 の場合、計算される出力が 127 より大きい場合 127 に、 または -128 より小さい場合 -128 になります。

パラメータとダ イアログボック ス

Block Paramete 🐒 Gain 🛛 💌
Gain
Element-wise gain ($y = K$.*u) or matrix gain ($y = K$ *u or $y = u$ *K).
Parameters
Multiplication: K.*u
Gain:
1
Saturate on integer overflow
OK Cancel Help Apply

Multiplication

入力の乗算に利用するタイプを指定します

•K.*u(要素毎の乗算)

•K*u(ゲインを左オペランドとする行列乗算)

•u*K (ゲインを右オペランドとする行列乗算)

Gain

スカラ、ベクトル、行列、変数名または式で表現されるゲイン。デフォルトは1。設定されていない場合、Gain パラメータのデータタイプは double です。Grain パラメータの値が長すぎてブロックに表示できない場合や要素毎の乗算が選択されている場合は、文字列-K-が表示されます。

Saturate on integer overflow

このオプションは、要素毎の乗算に対してのみ利用可能です。選択すると、 Gain ブロックの出力が整数オーバフローの原因となります。特に、出力デー タタイプが整数タイプの場合、出力タイプにより表現される最大値かまたは 計算された出力に関して、絶対値の意味で小さいほうの値をブロック出力と します。オプションを選択しない場合、Simulink は、Simulation Parameters ダイアログの Diagnostics ページの Data overflow イベントオプションで設定 された挙動を示します。(5-25 ページの "Diagnostics ページ"を参照)。 特性

直接フィードスルーありサンプル時間接続されるブロックから継承スカラ拡張入力および Gain パラメータについて状態0次元化可ゼロクロッシングなし

目的 ブロック入力を From ブロックに渡します。

ライブラリ Signals & Systems

詳細

XA

Goto ブロックは、その入力を対応する From ブロックに渡します。入力は、任意 のデータタイプの実数値または複素数値信号またはベクトルでも構いません。 From ブロックと Goto ブロックを用いると、実際にブロックを接続しなくても、 ブロック間で信号を渡すことができます。

From ブロックは、入力信号を複数の From ブロックに渡すことができますが、1 つの Goto ブロックからの信号しか受け取ることができません。Goto ブロックに 対する入力は、それに関連付けられた From ブロックに、それらのブロックが物 理的に接続されているかのように渡されます。From ブロックと Goto ブロックの 使用上の制限については、9-97 ページの From を参照して下さい。Goto ブロック と From ブロックは、Tag パラメータとして定義される Goto タグを用いること によって対応づけされます。

Tag visibility パラメータは、信号にアクセスする From ブロックの位置を制限するかどうかを決定します。

- local、これはデフォルトで、タグを用いる From ブロックと Goto ブロックは同じ サブシステム内になければなりません。local タグ名は、角括弧([])で囲みま す。
- scoped、これは、同じタグを用いる From ブロックと Goto ブロックが、同じサブシステム内か、モデル階層の Goto Tag Visibility ブロックの下のサブシステム内になければなりません。scoped タグ名は、中括弧 ({}) で囲みます。
- global、これは、同じタグを用いる From ブロックと Goto ブロックを、モデル内の任意の場所に置くことができます。

注意 マスクされたシステム内の scoped Goto ブロックは、そのサブシステム内と そのブロックが含むサブシステム内でのみ可視です。Simulink は、マスクされた サブシステム内の対応する scoped Goto ブロックよりもブロック線図内で高いレ ベルの Goto Tag Visibility ブロックをもつ図を実行または更新する場合には、エ ラーを発生します。

同じタグ名を用いる Goto ブロックと From ブロックが同じサブシステム内にあ る場合は、local タグを使ってください。同じタグ名を用いる Goto ブロックと From ブロックが異なるサブシステム内にある場合は、global または scoped タグ を使用しなければなりません。タグを global と定義すると、そのタグを使って同 じ信号にアクセスします。scoped に定義したタグは、モデル内の複数の場所で 使用することができます。つぎの例は、同じ名前 (A) をもつ 2 つの scoped タグ を使ったモデルを示します。



サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Goto ブロックは、	任意のデータタイプの実数または複素数信号を受け入れます。

Block Parameters: Go	oto		E
Goto			
Send signals to Fror 'scoped', then a Go visibility of the tag. tags are enclosed in braces, {}.	n blocks that hav to Tag Visibility bl The block icon di brackets, [], and	e the specified ta ock must be used splays the selecte scoped tag name	g. If tag visibility is I to define the ed tag name (local es are enclosed in
- Parameters			
Tag:			
A			
Tag visibility: local			
OK	Cancel	<u>H</u> elp	Apply

Tag

Goto ブロック識別子。このパラメータは、このブロック内で定義される範囲の Goto ブロックを識別します。

Tag visibility

Goto ブロックタグの範囲: local, scoped, global があります。デフォルトは、 local です。

特性

サンプル時間 接続されるブロックから継承

次元化

可

Goto Tag Visibility

目的 Goto ブロックタグの範囲を定義します。

ライブラリ Signals & Systems

 詳細
 Goto Tag Visibility ブロックは、scoped をもつ Goto ブロックタグのアクセス可能 なサブシステムを定義します。Goto tag パラメータとして指定されたタグは、 Goto Tag Visibility ブロックを含む同じサブシステム内と、モデル階層下のサブシ ステム内にある From ブロックからアクセスすることができます。

> Goto Tag Visibility ブロックは、Tag visibility パラメータ値が scoped に設定されて いる Goto ブロックに必要です。タグの可視化が local または global の場合は、こ のブロックは使用しません。ブロックアイコンは、中括弧 ({}) で囲んだタグ名 を示します。

サポートされて適用できるものはありません。 いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Goto Tag Visibility	×
GotoTagVisibility	
Used in conjunction with Goto and From blocks to define the visibility of scoped tags. For example, if this block resides in a subsystem (or root system) called MYSYS, then the tag is visible to From blocks that reside in MYSYS or in subsystems of MYSYS.	
Parameters	
Goto tag:	
A	
OK Cancel <u>H</u> elp <u>Apply</u>	

Goto tag

Goto ブロックタグの可視化がこのブロックの位置により決まります。

 サンプル時間
 N/A

 次元化
 N/A

特性

目的 未接続の入力端子を接地します。

ライブラリ Signals & Systems

詳細 Ground ブロックは、入力端子が他のブロックに接続されていないブロックと接 続するために使用できます。未接続の入力端子をもつブロックのあるシミュレー ションを実行すると、Simulink はワーニングメッセージを表示します。それらの ブロックを " 接地 " するために Ground ブロックを使用すると、 ワーニングメッ セージは回避できます。Ground ブロックは、ゼロ値の信号を出力します。信号 のデータタイプは、結合する端子をもつブロックと同じものです。

サポートされて Ground ブロックは、接続している同じ数値タイプ(実数または複素数)やデータ タイプと同じ信号を出力します。たとえば、つぎのモデルを考えましょう。. いるデータタイ プ



この例題において、Constant ブロックの出力は、Ground ブロックが接続してい る端子のデータタイプ (int8) を決めます。一方、この端子は、Ground ブロックに より出力される信号のタイプを決定します。



次元化

可

目的 クロッシングポイントを検出します。

ライブラリ Signals & Systems

詳細



Hit Crossing ブロックは、Hit crossing direction パラメータで指定した方向で、入 力が Hit crossing offset パラメータ値に到達する瞬間を検出します。

ブロックは、double のデータタイプの 1 入力を受け付けます。Show output port チェックボックスを選択すると、ブロック出力はクロッシングが発生する瞬間を 示します。入力信号がヒットクロッシングが検出された後のオフセット値と正確 に一致する場合、ブロックは値1の出力を続けます。隣接する2つの点での入力 信号がオフセット値を間に挟む場合(しかし、いずれの値もオフセットとは正確 に一致していない場合)、ブロックは2番目の時間ステップに対して値1を出力 します。Show output port チェックボックスを選択していない 場合、ブロックは シミュレーションによるクロッシングポイントの検出は行いますが、出力は生成 しません。

ブロックの Hit crossing direction パラメータが either に設定されているとき、ブ ロックは、有限数学およびコンピュータ精度の制限に対して機能するのに有効 な、"Almost Equal" ブロックとして提供されます。このような理由のために使用 すると、ブロックは条件を検出するための論理をモデルに追加するよりも便利な 場合があります。

デモ hardstop と clutch は、Hit Crossing ブロックの使い方を示します。hardstop デ モで、Hit Crossing ブロックは Friction Model サブシステムの中にあります。 clutch デモで、Hit Crossing ブロックは、Lockup Detection サブシステムの中にあ ります。

サポートされて Hit Crossing ブロックは、boolean の整合性モードが保たれる限り (4-49 ページの" いるデータタイ 厳密な Boolean 型の検査を有効にする"を参照)、boolean タイプの信号を出力し ます。ブロックは、double タイプの信号も出力します。 プ

パラメータとダ イアログボック ス	Block Parameters: Hit Crossing Hit Crossing Force simulation to locate ("hit") zero crossing of the input signal. Outputs 1 when hit crossing is detected, otherwise outputs 0.
	Parameters Hit crossing olfset: 0 Hit crossing direction: either ✓ Show output port OK Cancel Help Apply

Hit crossing offset

クロッシングを検出する値

Hit crossing direction

クロッシングを検出するために、入力信号がヒットクロッシングオフセット に近付いていく方向

Show output port

チェックすると出力端子を描画

特性

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	可
次元化	可
ゼロクロッシング	あり、クロッシングを検出するため

目的 信号の初期値を設定します。

ライブラリ Signals & Systems

詳細

IC ブロックは、その出力端子に接続されている信号の初期条件を設定します。

たとえば、つぎのブロックは、IC ブロックが "test signal" というラベルの付いた 信号をどのように初期化するかを示しています。



t=0 での信号値は3です。それ以降の信号値は6です。

IC ブロックは、ループの中の代数状態変数に対して、初期推定値を設定するときに有効なものです。詳細は、3-18 ページの"代数ループ"を参照してください。

IC ブロックは、double タイプの信号を受け入れ、出力します。

サポートされて いるデータタイ プ

ダイ	ア	ロク	゙ ゚ボッ
クス			

Block Parameters: IC	×
_ Initial Condition	
Initial condition for signal.	
Parameters	
Initial value:	
1	
OK Cancel <u>H</u> elp <u>Apply</u>	

Initial value

信号の初期値。デフォルトは1。

特性

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	パラメータのみ
状態	0

次元化 可 ゼロクロッシング なし 詳細

目的 サブシステムまたは外部入力に対する入力端子を作成します。

ライブラリ Signals & Systems

Inport は、システム外部からシステム内部へのリンクです。

Simulink は、つぎの規則に従って Inport ブロックの端子番号を割り当てます。

- 最上位システムまたはサブシステム内部のInport ブロックに、1から始まる連続的な番号を自動的に付けます。
- Inport ブロックを追加する場合、ブロックにはつぎに利用可能な番号が割り当てられます。
- Inport ブロックを削除する場合、その他の端子番号は自動的に番号を付け直され、その結果、Inport ブロックは連続的に、どの番号も省略されることはありません。
- Inport ブロックをシステムにコピーする場合、現在の番号がシステム内にすで にある Inport ブロックと一致しない限り、その端子番号に対して再度番号付け は行われません。コピーした Inport ブロックの端子番号が連続的でない場合に は、ブロックの番号を付け直さなければなりません。そうでないと、シミュ レーションの実行時またはブロック線図の更新時にエラーメッセージが表示 されます。

Inport ブロックに対する入力の次元を Port dimensions パラメータで指定するか、 または、-1 の値 (デフォルト)を指定することによって、Simulink に自動的に入 力の次元を決定させることができます。

Sample time パラメータは、信号がシステムに入力されるサンプルレートです。 デフォルト(-1)では、ブロックは、それに接続されているブロックからサンプル 時間を継承します。最上位システム内、またはサンプル時間を決定することがで きないブロックに Inport ブロックが接続されているようなモデル内では、Inport ブロックに対するこのパラメータを設定するとよい場合があります。

サブシステム内の Inport ブロック

サブシステム内の Inport ブロックは、サブシステムに対する入力を示します。 Subsystem ブロック上の入力端子に到達する信号は、そのサブシステム内の関連 する Inport ブロックから出力されます。

Subsystem ブロック上の入力端子に関連した Inport ブロックは、Subsystem ブ ロック上の入力端子の相対位置と一致する Port number パラメータをもつブロッ クです。たとえば、Port number パラメータが 1 であるブロックの場合、 Subsystem ブロックの一番上の端子と結線しているブロックからの信号を得ます。

Inport ブロックの Port number の番号を付け直すと、ブロックは、サブシステム 外部の同じブロックから信号を引き続き受け取りますが、異なる入力端子に接続 されます。

Inport ブロック名は、端子ラベルとして Subsystem ブロックアイコン上に表示されます。ラベルを表示しないようにするには、Inport ブロックを選択し、Format メニューから Hide Name を選択します。そして、Edit メニューから Update Diagram を選択します。

最上位システム内の Inport ブロック

最上位システム内の Inport ブロックには 2 つの使用法があります。1 つは、ワークスペースから外部入力を提供することで、これは Simulation Parameters ダイアログボックスか sim コマンドを使って行えます。解析関数で、モデルに摂動を加える手段を提供することです。

- ワークスペース内に存在するものを外部入力として使うためには、Simulation Parameters ダイアログ(5-19ページの"ベースワークスペースから入力の読み込み")を使うか、sim コマンド(5-38ページの sim を参照)のut 引数を使います。
- linmod や trim 解析関数により、モデルの摂動に関する情報を与えることができます。Inport ブロックは、入力がシステム内に取り込まれる点を定義します。解析コマンドとInport ブロックの使用法については、第5章を参照してください。
- サポートされて Inportは、任意のデータタイプの実数または複素数値信号を受け入れます。
 Inportの出力データタイプ、数値タイプは、対応する入力のデータタイプ、数値 クイプと同じです。InportのSignal typeパラメータと Data type パラメータを 使って、外部(例ワークスペース)入力の信号タイプ、データタイプをルートレ ベルの Inport に設定しなければなりません。

ルートレベルの Inport に接続する信号レベルの要素は、同じ数値タイプ、データ タイプでなければなりません。サブシステムの Inport に接続している信号要素 は、ある一つの例外を除いて、数値タイプ、データタイプが異なっていても構い ません。サブシステムが Enable ブロックまたは Trigger ブロックを含み、Inport が Outport と直接接続されている場合、入力要素は同じタイプでなければなりません。たとえば、つぎの Enabled サブシステムを考えます。



この例題で、In1に接続している信号ベクトル要素は、同じタイプでなければなりません。しかし、In2に接続している要素は、違ったタイプでも構いません。

パラメータとダ イアログボック ス

Block Parameters: In1	×
Inport	
Provide an input port for a subsystem or model. The 'Sample time' parameter may be used to specify the rate at which a signal enters the system.	
Parameters Port number:	
1	
Port dimensions (-1 for dynamically sized):	
-1	
Sample time:	
-1	
Data type: auto	
Signal type: auto	
☑ Interpolate data	
OK Cancel Help Apply	

Port number

Inport ブロックの端子番号

Port dimensions

Inport ブロックへの入力信号の次元。自動的に決定させるためには -1 を指定します。

Sample time

信号をシステムに入力するサンプリングレート

Data type

外部入力のデータタイプ

Signal type

外部入力の信号タイプ(実数または複素数)

注意 つぎの3つのパラメータは、ルートレベルの Inport にのみ適用されます。 サブシステムの Inport ダイアログには現れません。

Interpolate data

このオプションを選択すると、対応するワークスペースのデータが存在しない場合、対応する時間ステップで出力を内挿したり、外挿します。詳細は、 5-19ページの"ベースワークスペースから入力の読み込み"を参照してください。

特性

サンプル時間 接続されるブロックから継承

可

次元化

Integrator

目的 信号を積分します。

ライブラリ Continuous

Integrator ブロックは入力を積分します。積分器の出力は、単にその状態、つま り積分値です。Integrator ブロックを使って、つぎのことが行えます。

- ブロックダイアログボックス上で初期条件を定義するか、ブロックに対する 入力として、初期条件を定義できます。
- 状態を出力します。
- 積分の上限と下限を定義します。
- 追加のリセット入力を使って、状態をリセットします。

純粋な離散システムを作成する場合は Discrete-Time Integrator ブロックを使用してください。

初期条件の定義

初期条件は、ブロックダイアログボックス上でパラメータとして定義すること も、外部信号から入力することもできます。

- 初期条件をブロックパラメータとして定義するには、Initial condition source パ ラメータを internal として指定し、Initial condition パラメータフィールドに値 を入力します。
- 外部入力源から初期条件を提供するには、Initial condition source パラメータを external として指定します。下の図に示すように、ブロック入力の下に追加の 入力端子が表示されます。



状態端子の使用

つぎの既知の2つの状況のもとで、出力端子の代わりに状態端子を使う必要があ ります。

詳細



- ブロックの出力がリセット端子または初期条件端子によってブロックに フィードバックされ、代数ループを発生する場合。このような状況の例につ いては、bounce モデルを参照してください。
- 条件付きで実行されるサブシステムから別のサブシステムに状態を受け渡したい場合。この場合にはタイミングの問題が生じます。このような状況の例としては、clutchモデルを参照してください。

ユーザは、出力端子ではなく状態端子を通じて状態の受け渡しを行うことによ り、以上のような問題を回避することができます。たとえ、値が同一であって も、Simulink はそれらの数値をやや異なる時間で生成するため、これらの問題を 防ぐことができます。ブロックの状態は、Show state port チェックボックスを選 択することによって出力されます。デフォルトでは、図に示すように、状態端子 はブロックの上部に表示されます。



制限付きの積分

出力が指定可能なレベルを越えないようにするには、Limit output チェックボッ クスを選択し、該当するパラメータフィールドに限界値を入力します。そうする と、プロックは制限付き積分器と同じ機能をもちます。出力が限界値に近づく と、積分動作はオフとなり、積分が広がるのを防ぎます。シミュレーション中に 限界値を変更することはできますが、出力が制限付きかそうでないかの状態は変 更できません。出力はつぎのようにして決められます。

- 積分値が Lower saturation limit 未満で、入力が負の場合、出力は Lower saturation limit に保持されます。
- 積分値が Lower saturation limit と Upper saturation limit の間にある場合、出力 は積分値となります。
- 積分値が Upper saturation limit より大きくて、入力が正の場合、出力は Upper saturation limit に保持されます。

状態がいつ制限を受けるかを示す信号を生成するには、Show saturation port チェックブロックを選択します。下の図のように、飽和端子がブロック出力端子 の下に表示されます。



信号は、つぎの3つの値のいずれかです。

- 1は、上限が適用されたことを示します。
- 0は、積分が制限されないことを示します。
- -1 は、下限が適用されたことを示します。

このオプションを選択すると、ブロックには3つのゼロクロッシングが存在しま す。1つは飽和上限になるときを検出するもの、1つは飽和下限になるときを検 出するもの、そして、もう1つは飽和状態でなくなるときを検出するものです。

状態のリセット

ブロックは、外部信号に基づいて、指定した初期条件にその状態をリセットする ことができます。ブロックに状態をリセットさせるには、External resetの選択 肢の1つを選びます。つぎの図に示すように、ブロックの入力端子の下にトリガ 端子が表示されてトリガタイプを示します。



- risingを選択すると、トリガ信号が正の方向でゼロを横切る瞬間をトリガイベントとします。
- fallingを選択すると、トリガ信号が負の方向でゼロを横切る瞬間をトリガイベントとします。
- either を選択すると、トリガ信号が正または負いずれかの方向でゼロを横切る 瞬間をトリガイベントとします。
- levelを選択すると、リセットをトリガイベントとし、リセット信号が非ゼロであるあいだ、出力を初期条件に保持します。

リセット端子には直接フィードスルーがあります。ブロック出力がこの端子に直接、または直接フィードスルーをもつ一連のブロックを通してフィードバックされると、代数ループが発生します。このループを解除するには、代わりにブロッ

ク状態をリセット端子にフィードします。ブロックの状態にアクセスするには、 Show state port チェックボックスを選択します。

ブロックの状態に対して絶対許容値を指定

モデルに大きさが非常に異なる状態が含まれている場合、モデルに絶対許容値を 定義しても十分な誤差コントロールが得られない場合があります。Integrator ブ ロックの状態に絶対許容値を定義するには、Absolute tolerance パラメータに値 を与えます。ブロックに複数の状態がある場合は、すべての状態に同じ値が適用 されます。

誤差のコントロールに関する詳細は、5-14ページの"誤差許容値"を参照してくだ さい。

すべてのオプションの選択

すべてのオプションを選択すると、アイコンは、つぎのように表示されます。



サポートされて Integrator ブロックは、そのデータ端子上で double タイプの信号を受け入れ、出 **いるデータタイ** 力します。外部リセット端子は時間間隔 double や boolean タイプの信号を受け入 **パ** れます。

パラメータとダ イアログボック ス

Block Parameters: Inte	grator		×
- Integrator			
Continuous-time integ	gration of the inp	iut signal.	
Parameters			
External reset: no	one		•
Initial condition sou	urce: internal		_
Initial condition:			
0			
Limit output			
Upper saturation li	mit:		
inf			
Lower saturation li	mit		
-inf			
Show saturation	in port		
📃 🔲 Show state po	rt		
Absolute tolerance	:		
auto			
ОК	Cancel	Help	
	00.1001	<u></u> oip	CHAO.

External reset

リセット信号でトリガイベント (rising, falling, either, level) が発生すると、状態を初期条件にリセットします。

Initial condition source

Initial condition パラメータ (internal に設定した場合)、または外部ブロック (external に設定した場合)から、状態の初期条件を取り出します。

Initial condition

状態の初期条件。Initial condition source パラメータ値を internal に設定した ときに有効です。

Limit output

チェックすると、Lower saturation limit パラメータと Upper saturation limit パラメータの間の値に状態を制限します。

Upper saturation limit

積分の上限。デフォルトは inf。

Lower saturation limit

積分の下限。デフォルトは-inf。

Show saturation port

チェックすると、ブロックに飽和出力端子を追加します。

Show state port

チェックすると、ブロックの状態を出力するための端子をブロックに追加し ます。

Absolute tolerance

特性

ブロックの状態に対する絶対許容値。

直接フィードスルーあり、リセット端子および外部初期条件入力端子についてサンプル時間連続スカラ拡張パラメータについて状態接続されるブロックまたはパラメータから継承次元化可ゼロクロッシングLimit output オプションを選択すると、リセット検出のために1つ、飽和上限検出のためと飽和下限検出のためにそれぞれ1つ、そして飽和でなくなるときのために1つ

目的 入力に指定した論理演算を実行します。

ライブラリ Math

詳細



Logical Operator ブロックは、その入力につぎの論理演算を実行します。AND、 OR、NAND、NOR、XOR、およびNOTです。出力は、入力数、ベクトルサイ ズ、選択した演算子に依存します。TRUEの場合の出力は1で、FALSEの場合 の出力は0です。ブロックアイコンは、選択した演算子を示します。ブロックの 入力および出力にはつぎの規則が適用されます。

- ブロックが複数の入力をもつ場合、非スカラ入力は同じ大きさでなければなりません。たとえば、入力が2行2列の配列である場合、それ以外のすべての非スカラ入力も2行2列の配列でなければなりません。
- スカラ入力は、非スカラ入力と同じ大きさに拡張されます。
- ブロックが複数の入力をもつ場合、出力は入力と同じ大きさで(スカラ拡張後に)、各出力要素は、指定した論理演算子を対応する入力要素に適用した結果です。たとえば、指定した演算子がANDで、入力が2行2列の配列である場合、出力は、左上の要素が入力の左上の要素にANDを適用した結果で、他の要素についても同様である2行2列の配列です。
- ブロックが、単一入力でNOT演算子以外の演算子を指定する場合、入力はベクトルに似た形、つまり、スカラ、1次元配列、あるいは、1行あるいは1列の2次元配列にしなければなりません。出力は入力要素を演算した結果に等しいスカラ値です。
- 指定した演算子が NOT の場合、ブロックは 1 入力のみを受け入れます。出力 は、入力と同じ次元で、入力要素の論理補数を含んでいます。

多入力 XOR ゲートとして構成した場合、このブロックは、logic elements に対して IEEE 規格で決められているように、加算のモジュロ2演算を実行します。

 サポートされて いるデータタイ
 しogical Operator ブロックは、その入力端子で boolean 論理信号が利用可能な場合 (4-49 ページの " 厳密な Boolean 型の検査を有効にする " を参照)、boolean タイプ の信号のみ受け入れます。そして、ブロックは、double タイプの入力も受け入れ ます。double タイプの非零入力は、TRUE(1) として、零入力は FALSE(0) として 取り扱われます。すべての入力は、同じタイプでなければなりません。プロック の出力は、入力と同じタイプになります。 パラメータとダ イアログボック ス

Block Parameters: Logical Operator	×
C Logical Operator	
Logical operators. For a single input, operators are applied across the input vector. For multiple inputs, operators are applied across the inputs.	
Parameters	5
Operator: AND	
Number of input ports:	
2	
OK Cancel <u>H</u> elp <u>A</u> pply	

Operator

ブロック入力に適用される論理演算子。有効な選択肢は上記の演算子です。 Number of input ports

ブロック入力数。値は選択した演算子に適切なものでなければなりません。

特性

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	入力に関して
次元化	可
ゼロクロッシング	なし

詳細

目的 入力の区分的線形写像を実行します。

ライブラリ Functions & Tables

Look-Up Table ブロックは、ブロックのパラメータで定義した値の線形補間を 使って、入力を出力に写像します。

Vector of input values と Vector of output values パラメータを(行ベクトルまたは 列ベクトルとして)指定することによって、補間表を定義します。ブロックは、 ブロック入力を入力ベクトルの値と対応させることによって出力値を生成しま す。

- ブロックの入力と一致する値を見つけると、出力ベクトルの対応する要素が 出力されます。
- ・一致する値が見つからない場合は、補間表の適切な2つの要素間で線形補間を 実行し、出力値を決定します。ブロック入力が最初の入力ベクトル要素より 小さい場合、あるいは最後の入力ベクトル要素より大きい場合、ブロックは 最初の2つの点または最後の2つの点を使って外挿します。

2 つの入力を1 つの出力に写像するためには、Look-Up Table (2-D) ブロックを 使ってください。詳細は、9-135 ページの Look-Up Table (2-D) を参照してください。

ステップ状の遷移をもつ補間表を作成するには、異なる出力値をもつ入力値を繰 り返します。たとえば、つぎの入力および出力パラメータ値は、下のプロットに 示されるような入出力関係を生成します。

入力値のベクトル: [-2-1-1 000112] 出力値のベクトル values: [-1-1-2-212211]



この例は、u = -1, 0, +1 の 3 箇所にステップ状の不連続点があります。
与えられた入力値に2つの点がある場合、ブロックは、つぎの規則に従って出力 を生成します。

- *u*がゼロより大きい場合、出力は正の方向に原点から遠ざかるときに最初に検 出される点に関連した値を出力します。この例では、*u*が1のとき、*y*は黒丸 で印を付けた2となります。

Look-Up Table ブロックのアイコンは、入力ベクトルに対する出力ベクトルのグ ラフを表示します。ブロックのダイアログボックス上でパラメータを変更した場 合は、Apply または Close ボタンを押すとグラフが自動的に再描画されます。

サポートされて いるデータタイ プ Look-Up Table ブロックは、double タイプの信号を受け入れ、出力します。

パラメータとダ イアログボック ス

Block Parameters: Lo	ok-Up Table			×
Perform 1-D linear interpolation of input values using the specified table. Extrapolation is performed outside the table boundaries.				
Parameters				
Vector of input values:				
[-5:5]				
Vector of output values:				
tanh([-5:5])				
OK	Cancel	Help	Apply	

Vector of input values

可能なブロック入力値を含む値からなるベクトル。このベクトルは、出力ベ クトルと同じサイズでなければなりません。入力ベクトルは単調増加でなけ ればなりません。

Vector of output values

ブロック出力値を含む値からなるベクトル。このベクトルは、入力ベクトル と同じサイズでなければなりません。

直接フィー	ドスルー	あり

接続されるブロックから継承
不可
可
なし

目的 2入力の区分的線形写像を実行します。

ライブラリ Functions & Tables

詳細

Ŕ

Look-Up Table (2-D) ブロックは、ブロックのパラメータで定義した表の値の線形 補間を使ってブロック入力を出力に写像します。

可能な出力値は、Table パラメータとして定義します。その行と列に対応する値は、Row パラメータと Column パラメータで定義します。ブロックは、ブロック入力を Row パラメータおよび Column パラメータと比較することによって出力値を生成します。つぎの図に示すように、最初の入力は行に、2番目の入力は列に対応します。



ブロックは、入力値に基づいて出力を生成します。

- 入力が行パラメータ値と列パラメータ値に一致する場合、出力は行と列の交点における表の値になります。
- 入力が行パラメータ値と列パラメータ値に一致しない場合、ブロックは適切 な表の値の間で線形補間を行うことによって出力を生成します。ブロック入 力の少なくともどちらか一方が最初の行または列パラメータ値より小さい場 合、あるいは最後の行または列パラメータ値より大きい場合、ブロックは最 初の2つの点または最後の2つの点を用いて外挿します。

Row パラメータまたは Column パラメータのいずれかが反復値をもつ場合、ブロックは Look-Up Table ブロックについて説明した手法を用いて値を選択します。

Look-Up Table ブロックでは、単入力値を出力値のベクトルに写像することができます (9-132 ページの Look-Up Table を参照)。

例題

この例題では、ブロックパラメータは、つぎのように定義します。

Row: [1 2] Column: [3 4] Table: [10 20; 30 40] 最初の図は、行と列の値に一致するブロック入力の交点の値を出力するブロック を示しています。最初の入力は1で、2番目の入力は4です。これらの値は、最 初の行(行パラメータ値1)と2番目の列(列パラメータ値4)の交点で表の値を 選択します。



2番目の図で、最初の入力は1.7、2番目の入力は3.4です。これらの値によって、プロックは、左側のテーブルに示すように、行と列の値の間で内挿を行います。交点(28)の値が出力値です。



サポートされて Look-Up Table (2-D) ブロックは、タイプの信号を受け入れ、出力します。double いるデータタイ タイプの信号を受け入れ、出力します。

パラメータとダ イアログボック ス

プ

Block Parameters: Look-Up Table (2-D)	×			
Clookup Table (2-D) (mask) (link)				
Performs 2-D linear interpolation of input values using the specified input/output table. Extrapolation is performed outside the table boundaries.				
Parameters				
Row:				
[1:3]				
Column:				
[1:3]				
Table:				
[4 5 6;16 19 20;10 18 23]				
OK Cancel <u>H</u> elp <u>Apply</u>				

Row

ベクトルとして入力された、表に対する行の値。ベクトル値は単調増加でな ければなりません。 Column

ベクトルとして入力された、表に対する列ベクトル。ベクトル値は単調増加 でなければなりません。

Table

出力値の表。行列のサイズは、Row パラメータと Column パラメータで定義 した大きさと一致しなければなりません。

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	1 つの入力がベクトルの場合のもう1 つの入力について
次元化	可
ゼロクロッシング	なし

N 個の入力の定数、線形、スプライン内挿写像を行い、N 変数の関数のサンプル 化された表現を作成

ライブラリ Functions & Tables

詳細



Look-Up Table (n-D) ブロックは、関数 F が実験的にのみ既知であっても、T サン プルの内挿により N 変数関数のサンプル化された表現を実行し、 y = F(x1, x2, x3, ..., xn) に対する近似値を求めます。ブロックは、ブロックパ ラメータに V おって定義された値の表における内挿を使って、ブロック入力を

- 出力値に効率的に写像します。サポートされている内挿法は、以下の通りです。
 - Flat (定数)
 - 線形
 - Natural (キュービック) スプライン

これらの方法のいずれかを1次元、2次元、3次元および高次の表に適用することができます。

出力値を Table data パラメータとして定義し、行、列、N 次のブレークポイント 設定パラメータをもつ次元を定義します。ブロックは、ブロック入力とブレーク ポイント設定パラメータを比較することによって出力値を生成します。つぎの図 に示すように、最初の入力は最初の次元(行)のプレークポイントに対応し、2 番目のブレークポイントが列に対応します。



ブロックは、入力値に基づき出力を生成します。

- 入力がブレークポイントパラメータ値と一致する場合は、出力は、行、列、 高次のブレークポイントの交点での表の値です。
- 入力が行および列のパラメータ値と一致しない場合は、ブロックは適切な表の値を内挿して出力を生成します。ブロック入力の少なくともどちらか一方が最初の行または列パラメータ値より小さい場合、あるいは最後の行または列パラメータ値より大きい場合、ブロックはその次元でのブレークポイントの範囲に入力値を制限するか、外挿が可能な場合は線形外挿するか(キュー

ビックスプライン内挿が選択されている場合は)キュービック多項式を使って 外挿します。

注意 Look-Up Table(n-D) ブロックを使って PreLook-Up Index Search ブロックとと もに使うと、ある種の状況での線形内挿に対して、より柔軟性があり高性能にな ります。

非内挿ルックアップに対しては、ルックアップ操作が内挿の実行の代わりに簡単 な配列アクセスであるときに、Direct Look-Up Table, (n-D) ブロックを使って整数 値 k を得るので表の k 番目の要素だけが必要となります。y = table(k).

 サポートされて n-D Interpolated Look-Up Table ブロックは、double または single タイプの信号を受 け取りますが、n-D Interpolated Look-Up Table ブロックに対しては、入力はすべ プ

 プ

 プ

 プ

 プ

 アーグルのデータと Breakpoint 設定パラ メータは、入力と同じタイプでなければなりません。出力データタイプも入力 データタイプに設定されます。

パラメータとダ イアログボック ス

Block Parameters: Look-Up Table (n-D) 🛛 🛛 🔀			
LookupNDInterp (mask) (link)			
Perform n-dimensional interpolated table lookup including index searches. The table is a sampled representation of a function in N variables. Breakpoint sets relate the input values to positions in the table.			
Parameters			
Number of table dimensions: 2			
First input (row) breakpoint set:			
[10,22,31]			
Second (column) input breakpoint set:			
[10,22,31]			
Index search method: Binary Search			
E Begin index searches using previous index results			
Use one (vector) input port instead of N ports			
Table data:			
[4 5 6;16 19 20;10 18 23]			
Interpolation method: Linear			
Extrapolation method: None - Clip			
Action for out of range input: Warning			
OK Cancel <u>Help</u> Apply			

Number of table dimensions

Table data パラメータがとる次元数。これは、テーブルに対する独立変数の 数と、ブロックの入力数を決めます(下記の "Explicit Number of dimensions" と "Use one (vector) input port instead of N ports" を参照)。

First input (row) breakpoint set

ベクトルとして入力されるテーブルに対する行の値。ベクトルの値は、単調 増加でなければなりません。このフィールドは常に可視です。

Second (column) input breakpoint set

ベクトルとして入力されるテーブルに対する列の値。ベクトルの値は、単調 増加でなければなりません。このフィールドは、Number of table dimensions ポップアップが 2, 3, 4 または More の場合は可視です。

Third ... Nth input breakpoint set

ベクトルとして入力されるテーブルの3次元に対応する値。ベクトルの値 は、単調増加でなければなりません。このフィールドは、Number of table dimensions が3,4 または More の場合は可視です。

Fourth input breakpoint set

ベクトルとして入力されるテーブルの4次元に対応する値。ベクトルの値 は、単調増加でなければなりません。このフィールドは、Number of table dimensions が4または More の場合は可視です。

Fifth..Nth input breakpoint sets (cell array)

ベクトルからなる1次元セル配列として入力される、テーブルに対する3,4, またはそれ以上の次元に対応する値のセル。たとえば、{[10:10:30], [0:10:100]} は、5次および6次のブレークポイントに対して用いられる2つ のベクトルのセル配列です。ベクトルの値は、単調増加でなければなりませ ん。このフィールドは、Number of table dimensions が More の場合は可視で す。

Explicit number of dimensions

数が5以上のときのテーブルの次元数。Numbor of table dimensions が More の 場合に可視です。

Index search method

"Evenly Spaced Points", "Linear Search", "Binary Search"(デフォルト)から選択 します。各検索法は、状況に応じて、他のものよりも高速です。次善のイン デックス検索法の選択は、ルックアップテーブルに過度に依存してモデルの 速度を遅くする可能性があります。ブレークポイントデータが 10, 20, 30, ..., のように等間隔ならは、"Evenly Spaced Points" を選択することによってテー ブルのインデックスを直接計算するための最高の速度を得ることができま す。不定間隔のブレークポイントに対しては、入力信号が時間ステップの間 であまり変わらない場合には、同じ時間において "Linear Search" および "Begin index searches using previous index results" を使うことによって、最高の 速度になります。時間ステップ毎に1または2つのテーブル以上をジャンプ する、入力信号が急激に変化する不定間隔のブレークポイントに対しては、 "Binary Search" を選択することによって、最高の速度になります。"Evenly Spaced Points" アルゴリズムは、オフセットと残りの点の間隔の決定において 最初の2つのブレークポイントのみを利用します。

Begin index searches using previous index results

このオプションをアクティブにすると、ブロックは、前の時間ステップでの インデックスを使ってインデックス検索を初期化します。これは、入力信号 がある時間ステップからつぎのステップの間でテーブルにおいて位置が変更 されないときには、ブロックに対する性能を大きく向上させます。このオプ ションが非アクティブであるときは、線形検索とバイナリ検索法は、特に大 きいブレークポイントデータに対して、より時間がかかります。

Use one (vector) input port instead of N ports

独立変数毎に1つの入力をもつ代わりに、ブロックはN次元テーブルに対してN要素の幅である信号を予想する1つの入力端子を構成します。これは、大きNテーブルをもつブロック線図上でのラインのクラッタの除去に役立つことがあります。

Table data

出力値のテーブル。行列の大きさは、N breakpoint set パラメータまたは大き さが4を超えるときは Explicit number of dimensions パラメータによって定 義される大きさと一致しなければなりません。

Interpolation method

None (flat), Linear, Cubic Spline.

Extrapolation method

None (clip), Linear, Cubic Spline.

Action for out of range input

None, Warning, Error。シミュレーション中に範囲外の状態がある場合は、 "Warning" が選択されている場合はコマンドウィンドウにワーニングメッ セージが表示され、"Error" が選択されている場合はエラーメッセージと共に シミュレーションは停止します。

あり
接続されるブロックから継承
不可
不可
なし

目的 大きさ及び/または位相角信号を複素数信号に変換します。

ライブラリ Math

詳細



Magnitude-Angle to Complex ブロックは、大きさ及び / または位相角信号を複素 数値出力信号に変換します。入力は、double タイプの実数値信号でなくてはなり ません。角度入力はラディアン単位です。複素数値出力信号のデータタイプは double です。

入力は、双方共等しいサイズのベクトルである場合もあれば、1つの入力がベク トルでもう一方の入力はスカラとなる場合もあります。ベクトル入力の場合、出 力は複素数信号ベクトルとなります。大きさの入力ベクトルの要素は、対応する 複素数値出力要素の大きさに写像されます。角度入力ベクトルも同様に、複素数 値出力信号の角度に写像されます。入力がスカラの場合、入力は全ての複素数値 の出力信号の対応する成分(大きさまたは位相)に写像されます。

サポートされて いるデータタイ プ

パラメータとダ イアログボック

ス

上の詳細を参照してください。

DIUCK Farameters.	Maymuue-Ang	le to complex		
Magnitude-Angle to	Complex			
Construct a complex	output from magni	itude and/or radi	ian phase angle in	put.
- Parameters				
Innut Discusiture	AndAnalo			
Magnitude:				
0				
1				
	1	1	1	1
	O - 11		A second se	

Input

入力の種類を指定します。大きさ入力、角度入力またはその双方のいずれか を選択できます。

Angle (Magnitude)

入力が角度信号のみの場合、出力信号の大きさは一定になります。入力が大きさのみの場合、出力信号のラジアン単位の位相は一定になります。

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	関数が2つの入力を必要とするときの1つの入力
次元化	可
ゼロクロッシング	なし

目的 2つの入力間で切り替えを行います。

ライブラリ Nonlinear

詳細 Manual Switch ブロックは、2つの入力のどちらを出力へ受け渡すのかを選択す る切り換えスイッチの役割を果たします。入力の切り替えを行うには、ブロック アイコン上をダブルクリックしてください(このプロックはダイアログボックス をもちません)。選択された入力が出力へと受け渡され、選択されなかった入力 は無視されます。ユーザは、シミュレーション開始前にスイッチを設定したり、 シミュレーション実行中に信号の流れを対話的に制御するためにスイッチを切り 替えることができます。Manual Switch ブロックはモデルがセーブされる場合、 セーブ時点での状態を保持します。

- サポートされて
 Manual Switch ブロックは、全ての入力タイプを受け入れます。ただし、双方の

 いるデータタイ
 入力共に同一の信号タイプ及びデータタイプでなくてはなりません。このブロックの出力は、入力と同じ数値タイプ(実数または複素数)とデータタイプになります。
- **パラメータとダ**なし イアログボック ス

直接ノイードスルー	<i>U</i> ¹ <i>C</i> ¹
サンプル時間	接続されるブロックから継承
スカラ拡張	N/A
次元化	可
ゼロクロッシング	なし

Math Function

目的 数学関数を実行します。

ライブラリ Math

詳細

Math Function ブロックは、種々の一般的な数学関数を実行します。

ユーザは、つぎの Function リストから 1 つの関数を選択できます。exp, log, 10u, log10, square, sqrt, pow, reciprocal, hypot, rem, mod, transpose, hermitian が選択できる 関数です。ブロック出力は、単数または複数の入力における関数の結果です。

関数名がブロックアイコン上に表示されます。Simulink は、適切な数の入力端子 を自動的に描画します。

Fcn ブロックは、スカラ出力した生成しないので、ベクトル出力または行列出力 が必要な場合は、Math Function ブロックを使用してください。

サポートされて Math Function ブロックは、double タイプの複素数値または実数値の信号または 信号ベクトルを入力として受け入れます。出力信号タイプは、Output signal type いるデータタイ プ パラメータの設定にしたがって、実数または複素数になります。

ハフメータとタ	Block Parameters: Math Function	×
イアログボック ス	Math Mathematical functions including logarithmic, exponential, power, and moc functions.	ulus
	Function: exp	•
	Output signal type: auto	•
	OK Cancel Help Apply	

Function

数学関数を設定してください。

Output signal type

このダイアログボックスを使用すれば、Math Function ブロックの出力信号タイプを実数、複素数、自動選択のいずれかに設定することができます。

	入力	出力信号タイプ		ĵ
関数	信号	自動	実数	複素数
Exp, log, 10 ^u , log10, square, sqrt, pow, reciprocal, conjugate, transpose, hermitian	実数 複素数	実数 複素数	実数 エラー	複素数 複素数
magnitude squared	実数 複素数	実数 実数	実数 実数	複素数 複素数
hypot, rem, mod	実数 複素数	実数 エラー	実数 エラー	複素数 エラー

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	関数が2つの入力を必要とする場合の入力が対象
次元化	可
ゼロクロッシング	なし

目的 MATLAB 関数または式を入力に適用します。

ライブラリ Functions & Tables

詳細



MATLAB Fcn ブロックは、指定した MATLAB 関数または式を入力に適用しま す。設定された関数または式が入力に適用されます。関数の出力は、ブロックの 出力幅と一致していなければなりません。それ以外では、エラーになります。

以下に、このブロックに対する有効な式のいくつかの例を示します。

sin atan2(u(1), u(2)) u(1)^u(2)

注意 このブロックは、各積分ステップ中に MATLAB 文法解釈プログラムを呼 び出すので、Fcn ブロックよりも遅くなります。組み込みブロック (Fcn ブロッ クまたは Elementary Math ブロックなど)を代わりに使用するか、M-ファイルま たは MEX-ファイル S-ファンクションとして関数を作成し、S-Function ブロッ クを使ってそれにアクセスすることを検討してください。

サポートされて いるデータタイ プ MATLAB Fcn ブロックは、double タイプの複素数および実数値入力を受け入れ、 Output signal type パラメータの設定に従って、double タイプの複素数または実数 出力を生成します。

パラメータとダ イアログボック ス

Block Parameters: MATLAB Fcn 🛛
MATLAB Fon
Pass the input values to a MATLAB function for evaluation. The function must return a single vector argument of length 'Output width'. Examples: sin, sin(u), foo(u(1), u(2))
Parameters MATLAB function:
sin
Output dimensions:
-1
Output signal type: auto
Collapse 2-D results to 1-D
OK Cancel Help Apply

MATLAB function

関数または式。ある一つの関数のみを指定する場合は、入力引数を括弧に入れる必要はありません。

Output dimensions

出力次元。出力次元が入力次元と等しい場合、-1を指定します。それ以外の 場合は、正しい次元を指定する必要があります。そうでないとエラーを発生 します。

Output signal type

このダイアログボックスにより、MATLAB fcn の出力信号タイプを実数、複 素数、自動選択のいずれかに選択できます。自動選択の場合、このブロック の出力タイプは入力信号タイプと一致します。

Collapse 2-D results to 1-D

2次元配列を列優先の順番に2次元配列要素を並べた1次元配列として出力します。

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	N/A
次元化	可
ゼロクロッシング	なし

Matrix Gain

目的 入力と行列の乗算を実行します。

Gain ブロックを参照

ライブラリ Math

詳細 Matrix Gain ブロックは、行列ゲインを実現します。つぎのように、ベクトル入 力と指定した行列を乗算することによって出力を生成します。



サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Matrix Gain	×
Gain	
Element-wise gain ($y = K$.*u) or matrix gain ($y = K$ *u or $y = u$ *K).	
Parameters	
Gain:	
eye(3,3)	
Multiplication: Matrix(K*u)	
Saturate on integer overflow	
OK Cancel Help Apply	

Gain

行列として指定したゲイン。デフォルトは eye(3,3)。詳細は Gain ブロックを 参照してください。

Multiplication

信号とゲインの乗算に使用する乗法のタイプ。デフォルトは、行列の乗算で す。

Saturate on Integer Overflow

element-wise 乗法にのみ適用。詳細は Gain ブロックを参照してください。

特性	直接フィードスルー	あり
	サンプル時間	連続

スカラ拡張	不可
状態	0
次元化	可
ゼロクロッシング	なし

詳細

目的前回の積分ステップからのブロック入力を出力します。

ライブラリ Continuous

Memory ブロックは、その入力に対する1回分の積分ステップをサンプルアンド ホールドして、前の時間ステップからの入力を出力します。

つぎのサンプルモデル(より有効な情報を提供するためにさらに大きなモデルの 一部となる場合があります)は、シミュレーションで使用するステップサイズを 表示する方法を示します。Sum ブロックは、クロックで生成された現在の時間 から、Memory ブロックで生成された前のステップの時間を差し引きます。



注意 ブロックに対する入力が変化しない限り、ode15s または ode113 で積分を 行う場合、Memory ブロックの使用は避けて下さい。

サポートされて いるデータタイ プ Memory ブロックは、ユーザ定義のタイプを含んで、任意の数値タイプ(複素数 または実数)やデータタイプを受け入れます。入力タイプがユーザ定義の場合、 初期条件は0である必要があります。

パラメータとダ イアログボック ス

Block Parameters: Me	emory		×
- Memory			
Apply a one integrat	ion step delay. T	he output is the p	revious input value.
Parameters			
Initial condition:			
0			
🗖 Inherit sample	time		
OK	Cancel	<u>H</u> elp	Apply

Initial condition

初期積分ステップでの出力

Inherit sample time

チェックすると、サンプル時間は接続されるブロックから継承されます。

直接フィードスルー	なし
サンプル時間	連続、ただし、Inherit sample time チェックボックスを 選択すると継承
スカラ拡張	Initial condition パラメータについて
次元化	可
ゼロクロッシング	なし

目的 複数の入力ラインを1本のスカラ出力ラインに結合します。

ライブラリ Signals & Systems

詳細



Merge ブロックは、複数の入力を単一の出力ラインに結合し、任意の時点で、その出力ラインから出力される値は、接続されるプロックの最新の計算結果と等しくなります。ユーザは、Merge ブロックの Number of Inputs パラメータを使って、入力数を指定することができます。

注意 Merge ブロックは、交互に実行するサブシステムの作成を容易にします。 Merge の適用例については、適用例については、"交互に実行するサブシステム の作成"を参照してください。

Merge ブロックは、再配列された要素からなる信号を受け入れません。たとえば、つぎのダイアグラムの例では、



Selector ブロックによって、ベクトル信号の1番目の要素と4番目の入換えているため、この Merge ブロックは Selector ブロックの出力を受け入れることができません。

Merge ブロックの Allow unequal port widths オプションを選択しない場合、Merge ブロックは、等しい次元の入力のみを受け入れ、入力と等しい次元の信号を出力 します。Allow unequal port widths オプションが選択された場合、Merge ブロック



は、スカラ値かあるいは要素数が異なったベクトル(行列ではなく)を受け入れ

ます。さらに、各入力信号毎に出力信号の初まりに関係したオフセットを指定す ることができます。出力信号の幅は、 $max(w_1+o_1, w_2+o_2, \dots w_n+o_n)$ です。ここで、 $w_1, \dots w_n$ は入力信号の幅で、 $o_1, \dots o_n$ は入力信号のオフセット値です。たとえば、 つぎの図の Merge ブロックは、信号 v3 を作成するために信号 v1 と v2 を結合し ています。この例では、v1 のオフセットが 0、v2 のオフセットが 2 です。この 結果、6 要素幅の出力信号となります。Merge プロックは v1 の要素を v3 の最初 の 2 つの要素に写像し、v2 の要素を v3 の最後の 4 つの要素に写像します。

Initial Output パラメータを設定すれば、初期出力値を指定することができます。 Merge ブロックの初期出力を指定しない場合、接続されている複数のブロックに おいて初期出力が設定されていれば、Merge ブロックの初期出力は、接続されて いるブロックの初期出力を評価したものが出力されます。

 サポートされて
 Merge ブロックは、ユーザ定義のタイプを含む、任意の数値タイプ(複素数また

 いるデータタイ
 は実数)やデータタイプを受け入れます。入力タイプがユーザ定義の場合、初期

 プ
 条件は0 である必要があります。

パラメータとダ イアログボック ス

Block Parameters: Merge	×
Merge	
Merge the input signals into a single output signal whose initial value is specified by the "Initial output" parameter. If "Initial output" is empty, the Merge block outputs the initial output of one of its driving blocks.	
Parameters	
Number of inputs:	
E	
Initial output:	
0	
Allow unequal port widths	
Input port offsets:	
0	
OK Cancel Help Apply	

Number of inputs

マージする入力端子数。端子は、スカラまたはベクトルです。

Initial output

出力の初期値。指定されない場合、初期出力は、接続されうブロックのいず れか1つの初期出力と等価になります。

Allow unequal port widths

異なる要素数の入力を受け入れます。

Input port offsets

出力信号の始まりと関連する各々の入力信号のオフセット値を指定したベクトル。

特性

サンプル時間 接続されるブロックから継承 次元化 可 スカラ拡張 不可 目的 最小または最大の入力値を出力します。

ライブラリ Math

詳細

min

サポートされて

いるデータタイ

パラメータとダ

イアログボック

プ

ス

特性

MinMax ブロックは、入力の最小要素または最大要素を出力します。どちらの関 数を適用するかは、Function パラメータリストから 1 つを選択することができ ます。

ブロックに1つの入力端子がある場合、ブロックは入力ベクトルの最小要素また は最大要素であるスカラを出力します。

ブロックに複数の入力端子がある場合、非スカラー入力はすべて同じ次元にしな ければなりません。また、任意のスカラー入力は、同じ次元の非スカラー入力に 拡張されます。ブロックは、入力と同じ次元の信号を出力します。各々の出力要 素は、対応する入力要素の最小値あるいは最大値と等しくなります。

MinMax ブロックは、double タイプの実数値信号を受け入れ、出力します。

Block Parameters: MinM X MinMax Output min or max of input. For a single input, operators are applied across the input vector. For multiple inputs, operators are applied across the inputs. Parameters Function: max • Number of input ports: 11 0K Cancel <u>H</u>elp Apply

Function

入力に適用する関数 (min または max)

Number of input ports

ブロックに対する入力数

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	入力に関して

次元化 可 ゼロクロッシング あり、最小値と最大値を検出するため

目的 モデル内に作成に関する情報を表示します

ライブラリ Signals & Systems

詳細

Model Info

Annotation

Model Info ブロックは、モデルのブロック線図内に注釈ブロックとして、モデ ル作成の改訂情報を表示します。つぎの図は、vdp モデルに関する情報を表示す るために Model Info ブロックを用いた例です。



Model Info ブロックは、モデル自身の中に組み込まれている情報及び/または、 外部からの改訂情報やコンフィグレーション管理システムに関する情報を示しま す。Model Info ブロックのダイアログを使用して、Model Info ブロックによって 表示されるテキストの内容とフォーマットを指定します。

サポートされて 適用できるものはありません。 いるデータタイ

Model Info

ダイアログボッ

クス

Model Info: untitled _ 🗆 × Model properties: Editable text: Created Model Info --> Creator Annotation ModifiedBy ModifiedD ate ModifiedComment Model Version Description -Last Modification Date RCS properties: Author --> Date Revision Header Id. Locker RCSfile Source Horizontal text alignment: Center • Show block frame 0K Cancel Apply

Model Info ブロックダイアログボックスには、つぎの入力フィールドがあります。

Editable text. このフィールドには、Model Info ブロックによって表示するテキス トを入力します。ユーザは、%<propname>の形式で自由に変数を埋め込むこと ができます。ここで、propnameは、モデルまたは改訂コントロールシステムの プロパティ名をテキストで入力したものです。プロパティの値は、表示されたテ キスト内の対応する変数に置き換えられます。たとえば、モデルの現行のバー ジョンを 1.1 としましょう。このとき、つぎのテキストを入力すると

Version %<ModelVersion>

表示されたテキストには、つぎのように表示されます。

Version 1.1

ユーザが、このような方法で参照できるモデルと関連システムのプロパティは、 Model properties および Configuration manager properties フィールドにリスト アップされています。

Model properties. モデルに格納されている改訂コントロールのプロパティを表示 します。プロパティを選択し、矢印ボタンを選択すると対応する変数が Editable text フィールドに入力されます。たとえば、Created(作成日)を選択すると %<Created>% が Editable text フィールドに入力されます。4-114 ページの"バージョン管理プロパティ"を参照してください。

RCS properties. このフィールドは、このモデルに対して外部のコンフィグレー ションマネージャを前もって指定している場合にのみ表示されます(4-110ペー ジの "Configuration manager"を参照)。このフィールドのタイトルは、選択したコ ンフィグレーションマネージャを反映して変化します(たとえば、RCS Properties)。このフィールドは、Model Info ブロックに含めることができる外部 システムによって保持されるバージョン管理情報を利すと表示します。リストか ら項目を取り出すには、項目を選択し、隣接する矢印ボタンをクリックしてくだ さい。

注意 選択された項目は、コンフィグレーションマネージャによって保持されて いるモデルをチェックインまたはチェックアウトし、モデルをクローズして再度 オープンするまで Model Info ブロックの中に表示されません。 **目的** ブロック入力の選択

ライブラリ Nonlinear

詳細

Multiport Switch ブロックは、複数の入力からの選択を行います。

最初(先頭)の入力は制御入力で、その他の入力はスイッチ入力です。制御入力 の値によって、どのスイッチ入力が出力端子まで渡されるかが決まります。

制御入力が整数値でない場合は、Multiport Switch は値を最近傍の整数に丸め、 ワーニングを表示します。(丸められた)制御入力が1よりも小さいか、または 入力端子数よりも大きい場合は、スイッチは out-of-bounds エラーを表示します。 そうでない場合は、スイッチは対応するデータ入力を(丸められた)制御入力に 渡します。つぎの表は、Multiport Switch の挙動をまとめたものです。

(丸められた)制御入力	渡されるデータ
1 未満	Out of bound エラー
1	1 番目の入力
2	2 番目の入力
等々	等々
データ入力の数よりも大きい数 値	Out of bounds エラー

スイッチ入力はスカラでもベクトルでも構いません。制御入力もスカラでもベク トルでも構いません。ブロック出力は、つぎの規則で決まります。

- 入力がスカラの場合、出力はスカラとなります。
- ブロックに複数のスイッチ入力があり、少なくとも1つがベクトルの場合、出力はベクトルとなります。スカラ入力は、すべてベクトルに拡張されます。
- ブロックにスイッチ入力が1つだけあり、その入力がベクトルの場合、ブロックは制御入力を丸めた値に対応するベクトルの要素を出力します。

サポートされて
いるデータタイMultiport Switch の制御入力は、boolean 以外の任意の組み込みのデータタイプの
実数値信号を受け入れます。すべてのデータ入力は、同じデータタイプおよび数
値タイプでなければなりません。ブロック出力の信号タイプは、データ入力のタ
イプと同じです。

パラメータとダ イアログボック ス

Block Parameters: Multiport Switch					
Multi-Port Switch					
Pass through the input signals corresponding to the rounded value of the first input.					
Parameters					
Number of inputs:					
3					
OK Cancel <u>H</u> elp <u>Apply</u>					

Number of inputs

ブロックに対するデータ入力数

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	可
次元化	可
ゼロクロッシング	なし

目的 複数の入力信号を1つのベクトル信号または composite 出力信号にまとめます。

ライブラリ Signals & Systems

詳細

≯

Mux ブロックは、複数の入力を1つの出力にまとめます。入力は、スカラ、ベクトル、行列信号です。入力に従って、Mux ブロックの出力はベクトルまたは行列要素とベクトル要素の両方を含む信号である composite 信号です。Mux ブロックのすべての入力がベクトルまたはベクトルライクである場合は、ブロックの出力はベクトルです。ベクトルライクな信号とは、スカラ(1要素ベクトル)、ベクトル、1列または1行行列である信号のことです。入力が非ベクトルライクな行列信号である場合は、Mux の出力は、バス信号です。バス信号は、バーチャルブロック(例., Demux, Subsystem, Go To ブロック)のみに接続されます。

Mux ブロックの Number of Inputs パラメータを使って、入力信号の名前と次元、 入力数を指定することができます。つぎのフォーマットを使ってこのパラメータ を指定することができます Mux ブロックの Number of Inputs パラメータを使っ て、入力信号の名前と大きさ、入力数を指定することができます。つぎのフォー マットを使ってこのパラメータを指定することができます。

・スカラ

Mux ブロックへの入力数を指定します。このフォーマットが用いられるとき は、ブロックは任意の大きさの信号を受け入れます。また、Simulink は各々の 入力に名前 signalN を割り当てます。ここで、N は入力端子番号です。

・ベクトル

ベクトルの長さは入力の数を指定し、各々の要素は対応する入力の次元を指 定します。正の値は、対応する端子がそのサイズのベクトルだけを受け入れ ることを示しています。例えば、[23]は、サイズ2と3の二つ入力端子を定 義します。入力信号の幅が指定した幅と異なる場合、Simulinkは、エラーメッ セージを表示します。値-1は、対応する端子が、ベクトルや任意の次元の行 列を受け入れるように定義されます。

セル配列

セル配列の長さは、入力の数を指定します。各々のセルの値は、対応する入 力端子の次元を指定します。スカラー値Nは、Nサイズのベクトルを指定し ます。ベクトル値[MN]は、M×N行列を指定します。値-1は、対応する端 子が、任意の次元の信号を受け入れることを定義します。

・ 信号名のリスト

コンマで区切って信号名のリストを入力することができます。Simulink は、対応する端子と信号に各々の名前を割り当てます。たとえば、position,velocity と

入力すると、Mux ブロックは、position と velocity という名前の 2 入力になります。

注意 Simulink は、Simulink ブロックライブラリからモデルに Mux ブロックをコ ピーする場合には、ブロック名を非表示にします。

サポートされて Mux ブロックは、タイプが混在するベクトルを含め、任意のデータタイプの実 **いるデータタイ** 数値および複素数値信号を受け入れます。 プ

パラメータとダ イアログボック ス

Block Parameters: Mux					
Mux					
Combine scalar or vector signals into larger vectors.					
Parameters					
Number of inputs:					
2					
Display option: bar					
OK Cancel Help Apply					

Number of inputs

入力数と入力幅。このパラメータフィールドに対して信号名をカンマで区 切ったリストとして入力することができます。

Display option

ユーザのモデル内のブロックアイコンの表示法

表示オプション	モデル内のプロックの表示法	
none	Mux がブロックアイコン内部に表示	
signals	端子の側に信号名を表示	
bar	はっきりとした棒のアイコンが表示	

Outport

詳細

1

目的 サブシステムまたは外部出力のための出力端子を作成します。

ライブラリ Signals & Systems

Outport は、システムからシステム外部へのリンクです。

Simulink は、つぎの規則に従って Outport ブロックの端子番号を割り当てます。

- 最上位システム内またはサブシステム内のOutportブロックに1から連続した番号を自動的に付けます。
- Outport ブロックを追加する場合、ブロックにはつぎに使用可能な番号が割り 当てられます。
- Outport ブロックを削除する場合、その他の端子番号は自動的に番号が付け直され、その結果、Outport ブロックは連続的でどの番号も省略されることはありません。
- ・ Outport ブロックをシステムにコピーする場合、カレントの番号がシステム内にすでに存在する Outport ブロックと一致しない限り、その端子番号に対する番号を付け直しません。コピーした Outport ブロックの端子番号が連続的でない場合には、ブロックの番号付けをし直す必要があります。そうでないと、シミュレーションの実行時またはブロック線図の更新時にエラーメッセージが表示されます。

サブシステム内の Outport ブロック

サブシステム内の Outport ブロックは、サブシステムからの出力を表します。サ ブシステム内の Outport ブロックに到達する信号は、その Subsystem ブロック上 の関連する出力端子から出力されます。Subsystem ブロック上の出力端子に関連 付けられた Outport ブロックの Port number パラメータは、Subsystem ブロック 上の出力端子の相対的な位置と一致します。たとえば、Port number パラメータ が1である Outport ブロックは、Subsystem ブロック上の最上部の出力端子に接 続されたブロックにその信号を送ります。

Outport ブロックの Port number の番号を付け直すと、サブシステム外部の同じ ブロックに信号を送り続けるように(それに合った)異なる出力端子に接続され ます。

既存のブロックを選択することによってサブシステムを作成する場合、複数の Outport ブロックがグループ化されたブロックに含まれていると、Simulink は自 動的にブロック上の端子の番号を付け直します。 Outport ブロック名は、端子ラベルとして Subsystem ブロックアイコンに表示されます。ラベルを表示しないようにするためには、Outport ブロックを選択し、 Format メニューから Hide Name を選択してください。

条件付きで実行されるサブシステム内の Outport ブロック

Outport ブロックが Enabled サブシステム内にある場合、サブシステムが実行不能なときにその出力をどうするかを指定することができます。すなわち、出力を 初期値に reset するか、最新の値に held することができます。Output when disabled ポップアップメニューからこれらのオプションが提供されます。Initial output パラメータは、サブシステムが実行される前と、reset オプションが選択さ れた場合のサブシステムの実行不能な状態での出力値です。

最上位システム内の Outport ブロック

最上位システム内の Outport ブロックには 2 つの使用法があります。1 つは、 ワークスペースに対して外部出力を提供することで、これは Simulation Parameters ダイアログボックスまたは sim コマンドを使って実行することができ ます。もう 1 つは、解析関数に対してシステムから出力を得るための手段を提供 することです。

外部出力をワークスペースに出力するには、Simulation Parameters ダイアログボックス (5-22 ページの "Saving Output to the Workspace"を参照)または sim コマンド (5-38 ページの sim). たとえば、システムが複数の Outport ブロックをもっていると、つぎのコマンド

[t,x,y] = sim(...);

は、yを各列が種々のOutport ブロックに対してデータを含むように行列として出力します。列の順番は、Outport ブロックに対する端子番号の順番と一致します。

2 番目の(状態)引数の後に複数の変数名を指定した場合、各 Outport ブロッ クからのデータは異なる変数に書き出されます。たとえば、2 つの Outport ブ ロックがシステムにある場合、Outport ブロック 1 からのデータを speed に保 存し、Outport ブロック 2 からのデータを dist に保存するためには、つぎのよ うにコマンドを指定します。

[t,x,speed,dist] = sim(...);

 システムからの出力を得るため、解析関数 linmod と trim に対する方法を与え ます。解析コマンドを用いた Outport ブロックの使用法に関しては、第5章を 参照してください。 サポートされて いる数値タイプ とデータタイプ やものと同じです。Outport ブロックの出力の数値タイプ、データタイプは、入力 のものと同じです。Outport ブロックに接続されている信号ベクトルの要素は、 つぎの状況を例外として、異なる数値タイプ、データタイプでもかまいません。 outport ブロックが条件付きで実行されるサブシステム内にあり、初期出力が指 定されていない場合、入力ベクトルのすべての要素は、同じ数値タイプおよび データタイプでなければなりません。

> Simulinkのデータタイプ変換ルールは、outportのInitial output パラメータに適用されます。初期値がブロックの出力データタイプの範囲内である場合、 Simulink は初期値を出力データタイプに変換します。変換が精度の低下を伴う場合は、Simulink はワーニングメッセージを表示します。指定された初期出力が出力データタイプの範囲外である場合は、Simulink はシミュレーションを停止し、エラーを表示します。ブロックの出力データタイプは、その入力に接続されている信号のデータタイプです。

パラメー	-タと	゠ダ
イアログ	ブボッ	ック
ス		

Block Parameters: Out1 Outport	×			
Provide an output port for a subsystem or model. The 'Dutput when disabled' and 'Initial output' parameters only apply to conditionally executed subsystems. When a conditionally executed subsystem is disabled, the output is either held at its last value or set to the 'Initial output'. The 'Initial output' parameter can be specified as the empty matrix, [], in which case the initial output is equal to the output of the block feeding the output.				
Parameters Port number:				
Output when disabled: held				
Initial output: []				
OK Cancel <u>H</u> el	p <u>A</u> pply			

Port number

Outport ブロックの端子番号

Output when disabled

条件付きで実行されるサブシステムに対して、システムが利用不可能のとき のブロック出力の扱い方

Initial output

条件付きで実行されるサブシステムに対して、サブシステムの実行前とサブ システムが利用不可能であるときのブロック出力
特性
 サンプル時間
 接続されるブロックから継承

 次元化
 可

目的 要素毎の積または商、行列の積または逆行列演算を実行

ライブラリ Math

詳細

) | × | Product ブロックは、Multiplication および Number of inputs パラメータの値に応じて、入力の要素毎または行列の積を出力します。

 Number of inputs パラメータの値が、* と / 記号の組み合わせの場合、ブロック 入力の数は記号の数と等しくなります。ブロックアイコンには、各入力端子の隣に適切な記号を表示されます。

たとえば、*/をパラメータ値として入力すると、ブロックアイコンはつぎのようになります。



Multiplication パラメータは、element-wise です。

Multiplication パラメータの値が element-wise ならば、ブロック出力は*印が付けられたすべての入力を/印が付けられたすべての入力で割った、要素毎の積です。たとえば、入力がサイズnのベクトルである場合、出力は各要素が以下と等しいサイズnのベクトルです。

 $y_i = u1_i \times u2_i \times \ldots \times un_i$

(入力ベクトルの内積を作成するには、Dot Product ブロックを使います)。

入力が行列である場合は、すべての入力は行列またはスカラでなければなり ません。ここで、スカラは1行1列行列または1要素ベクトルとして定義さ れます。入力がベクトルである場合は、すべての入力はベクトルライクでな ければなりません。ベクトルライクな入力とは、スカラ、ベクトル、列行列 または行行列です。すべての非スカラ入力は、同じ大きさでなければなりま せん。入力は列行列と行行列の両方をもつことはできません。

Multiplication パラメータの値が matrix ならば、ブロック出力は、* 印が付けら れた入力と / 印が付けられた入力の逆行列を乗算した行列積です。演算の順番 は、Number of Inputs フィールドで指定された順番と同じです。たとえば、値 */* は行列積 AB-1C になります。ここで、A, B, C はそれぞれ 1,2,3 番目の入力 信号です。行列の大きさは、行列積が定義された大きさでなければなりません。

すべての入力がスカラである場合は、ブロックの出力もスカラです。そうで ない場合は、出力は、入力が行列かベクトルかに応じて、行列またはベクト ルになります。

 Number of inputs パラメータの値が*ならば Multiplication パラメータの値は element-wise で、入力はベクトルライク、つまり、1次元配列か1行あるいは 1列の2次元配列で、ブロックは入力要素のスカラ積を出力します。

 $y = \Pi u_i$

この場合、ブロックアイコンは、つぎのように表示されます。



入力が行列で、**Multiplication** パラメータが element-wise ならば、Simulink はエ ラーを表示します。**Multiplication** パラメータの値が matrix ならば、ブロック は入力を変更せずに出力します。

- Number of inputs パラメータの値が / ならば、Multiplication パラメータの値は element-wise で、入力は 1 次元配列で、ブロックは入力要素のスカラ積を逆に して出力します。入力が行列で、Multiplication パラメータが element-wise な らば、Simulink はエラーを表示します。Multiplication パラメータの値が matrix ならば、ブロックは、入力の逆行列を出力します。
- スカラ値を Number of inputs パラメータとして入力することは、*キャラクタの文字列を入力することと等価です。ここで、文字列の長さはスカラ値です。
- ブロックが入力を1つもつ場合は、スカラまたはベクトルでなければなりません。

必要ならば、Simulink はブロックをリサイズして、すべての入力端子を表示しま す。入力数が変更されたときは、端子はブロックの一番下から追加または削除さ れます。

サポートされて いるデータタイ プ Product ブロックは、要素毎の乗算に対しては、任意のデータタイプの実数また は複素数値信号を受け入れます。すべての入力信号は、同じデータタイプでなけ ればなりません。出力信号のデータタイプは、入力のタイプを同じです。入力 は、行列乗算に対しては、single または double タイプの実数または複素数値信号 でなければなりません。

パラメータとダ イアログボック ス

Block Parameters: Product	×	
Product 45		
Multiply or divide inputs. Choose element-wise or matrix product and specify one of the following: a) "or / for each input port (e.g., "*/") b) scalar specifies the number of input ports to be multiplied Scalar value of "1' for element-wise product causes all elements of a single input vector to be multiplied. If / is specified with matrix product, compute the inverse of the corresponding input.		
Parameters Multiplication: Element-wise(.*)		
Number of inputs:		
2		
Saturate on integer overflow		
OK Cancel Help Apply		

Multiplication

入力の積を作成するために、要素毎の乗算を用いるか、行列乗算を用いるか を指定します。

Number of inputs

ブロックに対する入力数または*と/記号の組み合わせ。デフォルトは2。

Saturate on integer overflow

このオプションは、要素毎の乗算に対してのみ利用可能です。選択すると、 このオプションは、Product ブロックの出力が整数オーバフローの原因とな ります。特に、出力データタイプが整数タイプの場合、出力タイプにより表 現される最大値か、または計算された出力に関して絶対値の意味で小さいほ うの値をブロック出力とします。選択しない場合は、Simulink は Simulation Parameters ダイアログの Diagnostics ページの Data overflow イベントオプ ションで設定された挙動を示します (5-25 ページの "Diagnostics ページ"を参 照)。

特性

直接フィードスルー あり

サンプル時間	接続されるブロックから継承
スカラ拡張	可
次元化	可
ゼロクロッシング	なし

目的 1 つのラインの幅、サンプル時間及び/または複素数信号であるかどうかについて調査します。

Probe ブロックは、double タイプの信号を受け入れ、出力します。

- ライブラリ Signals & Systems
- 詳細



Probe ブロックは、入力された信号について希望する情報を出力します。Probe ブロックは、入力信号の幅、次元、ンプル時間、そして、入力が複素信号である かを示すフラグを出力します。Probe ブロックは、1 つの入力端子をもちます。 一方、出力端子の数は、調査対象として何を選択したか(信号の幅、次元、サン プル時間、そして複素数信号であるかを示すフラグ)によって決まります。それ ぞれの調査結果は、別々の出力端子から別々の信号として出力されます。Probe ブロックには組み込まれた任意のデータタイプの実数値、複素数値の信号または 信号ベクトルが入力できます。ただし、出力される信号は double タイプです。 シミュレーション中、このブロックのアイコンは調査結果データを表示します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Probe	×
Probe	
Probe a line for its width, sample time, or complex signal flag.	
Parameters	
☑ Probe width	
Probe sample time	
Probe complex signal	
✓ Probe signal dimensions	
OK Cancel <u>H</u> elp <u>A</u> pply	

Probe width

チェックした場合、調査対象信号の出力幅(要素数)が出力されます。

Probe sample time

チェックした場合、調査対象ラインのサンプル時間が出力されます。

Probe complex signal

チェックした場合、調査対象の信号が複素数であれば1,そうでない場合に は0を出力します。 Probe signal dimensions

チェックした場合、調査対象信号の次元が出力されます。

特性

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	可
次元化	可
ゼロクロッシング	なし

目的 規則的な間隔でパルス波を発生します。

ライブラリ Sources

詳細



Pulse Generator ブロックは、規則的な間隔でスカラ、ベクトル行列のパルスを発生します。ブロックのパラメータ Amplitude, Period, Duty cycle, Start time は、出力信号の特性を決定します。スカラ拡張後にすべて同じ次元をもっていなければなりません。また同じデータと数値(複素数または実数)タイプでなければなりません。

連続時間システムに対して、Pulse Generator ブロックを使います。離散信号を生成するには、Discrete Pulse Generator ブロックを使います。

サポートされて Pulse Generator ブロックは、任意のデータタイプの実数または複素数信号を出力 いるデータタイ します。データと出力信号の数値(実数または複素数)タイプは、Amplitude パ プ ラメータと同じです。

パラメータとダ イアログボック ス

Block Parameters: Pulse Generator	×
Pulse Generator (mask) (link)	
Pulse Generator	
Parameters	
Period (secs):	
۵	
Duty cycle (% of period):	
50	
Amplitude:	
1	
Start time:	
0	
☑ Interpret vector parameters as 1-D	
OK Cancel Help Apply	

Period

秒単位のパルス周期。デフォルトは1秒です。

Duty cycle

デューティサイクル:信号がオンであるパルス周期の割合。デフォルトは 50 パーセントです。

Amplitude

パルス振幅。デフォルトは1。

Start time

パルス発生前の秒単位での遅れ。デフォルトは0秒です。

Interpret vector parameters as 1-D

このオプションを選択しスカラ拡張後のパラメータが1行または1列の行列 の場合、ブロックは1次元信号(ベクトル)を出力します。そうでない場合、 出力の次元は、パラメータの次元と同じになります。

特性 サンプル時間 継承 スカラ拡張 パラメータに関して 次元化 可 ゼロクロッシング なし 目的 指定した間隔で入力を離散化します。

ライブラリ Nonlinear

詳細



プ

Quantizer ブロックは、入力信号を階段状関数に渡し、入力軸上の多数の隣接点 が出力軸上の1点に写像されるようにします。その効果は、滑らかな信号を階段 状の出力に量子化することです。出力は、ゼロに関して対称な出力を生成する四 捨五入方式で計算されます。

y = q * round(u/q)

ここで、y は出力、u は入力、q は Quantization interval パラメータです。

×

サポートされて Quantizer ブロックは、single あるいは double タイプの実数または複素数信号を受 いるデータタイ け入れ、出力します。

Apply

パラメータとダ	Block Parameters: Quantizer
イアログボック	Quantizer Discretize input at given interval.
A	Parameters Quantization interval
	0.5
	✓ Treat as gain when linearizing

ΟK

Quantization interval

出力を量子化する間隔。Quantizer ブロックの許容出力値は n*q です。ここで、n は整数で q は Quantization interval です。デフォルトは 0.5 です。

Treat as gain when linearizing

Cancel

<u>H</u>elp

Simulink は、線形化する時に、デフォルトで、Quantizer ブロックをゲインとして扱います。これは、大きい信号の線形化問題です。このボックスを チェックしない場合、線形化ルーチンは、小さい信号問題を仮定し、ゲイン をゼロを設定します。

特性 直接フィードスルー あり サンプル時間 接続されるブロックから継承

Quantizer

スカラ拡張パラメータに関して次元化可ゼロクロッシングなし

目的 一定の割合で増加または減少する信号を発生します。

ライブラリ Sources

詳細 Ramp ブロックは、指定した時間と値で始まり、指定の変化率をもつ信号を発生します。ブロックは、出力信号の特性を決めるために Slope、Start time、Duty Cycle、Initial output パラメータをもちます。すべてのパラメータは、スカラ拡張した後、同じ次元でなければなりません。

Ramp ブロックは、double タイプの信号を出力します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

lock Parameters: Ra	amp			×
- namp (mask) (iink)				
ramp				
Parameters				
Slope:				
1				
Start time:				
0				
I livitial output				
miliai output.				
0				
Interpret ve	ctor parameters	as 1-D		
		1	1	

Slope

発生させる信号の変化率。デフォルトは1。

Start time

信号発生開始時間。デフォルトは 0。

Initial output

信号の初期値。デフォルトは 0。

Interpret vector parameters as 1-D

このオプションをチェックし、パラメータが1行あるいは1列の行列の場合、ス カラ拡張の後、ブロックは1-D信号(ベクトル)を出力します。そうでない場合 は、パラメータと同じ次元の信号を出力します。

特性	サンプル時間
	フカラ坊で

接続されるブロックから継承

スカラ拡張 可

次元化 可

ゼロクロッシング あり

目的 正規分布乱数を発生します。

ライブラリ Sources

詳細

M

レーションが開始されるごとに指定の値にリセットされます。 デフォルトで、生成される数列は、平均が0、分散が1になります。しかし、これらの値は、自由に変更することが可能です。このデータ列は、繰り返し可能で、同じシードとパラメータをもつ任意のRandom Number ブロックから発生させることができます。同じ平均と分散をもつ乱数ベクトルを発生するには、

Random Number ブロックは、正規分布乱数を発生します。シードは、シミュ

ー様分布する乱数を生成するには、Uniform Random Number ブロックを用いてく ださい。

Initial seed パラメータとベクトルとして指定します。

ソルバは比較的に滑らかな信号を積分するものなので、ランダムな信号を積分し ないようにしてください。代わりに、Band-Limited White Noise ブロックを使用 してください。

すべてのブロックの数値パラメータは、スカラ拡張後、同じ次元にならなければ なりません。

Random Number ブロックは、double タイプの信号を受け入れ、出力します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Random Number	×
Random Number	
Output a normally (Gaussian) distributed random signal. Output is repeatable for a given seed.	
Parameters	
Mean:	
Q	
Variance:	
1	
Initial seed:	
0	
Sample time:	
0	
Interpret vector parameters as 1-D	
OK Cancel <u>H</u> elp Apply	

Mean

乱数の平均で、デフォルトは0です。

Variance

乱数の分散で、デフォルトは1です。

Initial seed

乱数発生器に対する開始シード。デフォルトは0。

Sample time

サンプル間の時間間隔で、デフォルトは0です。この場合は、連続サンプル 時間になります。

Interpret vector parameters as 1-D

このオプションをチェックし、パラメータが1行あるいは1列の行列の場合、スカラ拡張の後、ブロックは1-D信号(ベクトル)を出力します。そうでない場合は、パラメータと同じ次元の信号を出力します。

特性

サンプル時間	連続または離散
スカラ拡張	パラメータに関して
次元化	可
ゼロクロッシング	なし

目的 信号の変化率を制限します。

ライブラリ Nonlinear

詳細



- Rate Limiter ブロックは、このブロックを通過する信号の一次微係数に制限を与えます。出力は、制限を与えない場合よりも遅くなります。微分は、つぎの方程式を使用して計算されます。 $rate = \frac{u(t) y(t-1)}{t(i) t(i-1)}$
- $u(i) \ge t(i)$ は現在のブロック入力と時間で、 $y(i-1) \ge t(i-1)$ は前のステップでの出力と時間です。出力は、*rate* を Rising slew rate パラメータおよび Falling slew rate パラメータと比較することによって決定されます。
- *rate* が Rising slew rate パラメータ (R) より大きい場合、出力はつぎのように計算されます。

 $y(i) = \Delta t \cdot R + y(i-1)$

rate が Falling slew rate パラメータ(F)より小さい場合、出力はつぎのように計算されます。

 $y(i) = \Delta t \cdot F + y(i-1)$

rate が *R* と *F* の範囲内にある場合、出力の変化は入力の変化に等しくなります。

Rate Limiter ブロックは、double タイプの信号を受け入れ、出力します。

y(i) = u(i)

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

llock Parameters: Ra - Rate Limiter	te Limiter			2
Limit rising and falling	g rates of signal.			
Parameters				
Rising slew rate:				
1				_
Falling slew rate:				
-1				
,				
OK	Canool	Halo	Applu	

Rising slew rate

増加する入力信号の微係数の範囲

Rate Limiter

Falling slew rate

減少する入力信号の微係数の範囲

特性

直接フィールドスルー	あり
サンプル時間	連続
スカラ拡張	入力およびパラメータについて
次元化	可
ゼロクロッシング	なし

目的 大きさと位相角信号を複素数信号に変換します。

換します。

す。

上の詳細を参照。

ライプラリ Math

詳細

>Re-___ >Im-__> 入力は、双方共等しいサイズをもつ配列(ベクトルまたは行列)である場合もあ れば、1つの入力が配列であるのに対して、もう一方の入力はスカラとなる場合 もあります。ブロックが配列の入力をもつ場合、出力は、同じ大きさの複素数配 列です。実数入力の要素は、対応する複素数出力要素の実部に写像されます。虚 数入力は、複素数出力信号の虚数部分に同様に写像されます。入力がスカラの場

合、すべての複素数値出力信号の対応する成分(実数または虚数)に写像されま

Real-Imag to Complex ブロックは、実数や虚数の入力を複素数値の出力信号に変

ブロックが実数入力と虚数入力をもつ場合、入力は、任意のデータタイプでかま いません。出力は、同じタイプになります。ブロックのパラメータが出力の複素 数部分または虚数部分を決める場合、入力は double である必要があり、出力は double です。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: I	Real-Imag to	Complex				×
Real-Imag To Comp Construct a complex	ilex output from real	and/or im	naginary ir	nput.		
Parameters						
Input: RealAndIm	ag		ſ			•
Imag part:						
0						
ОК	Cancel		<u>H</u> elp		Apply	

Input

入力の種類を指定します。実数入力、虚数入力またはその双方を選択できま す。 Real (Imag) part

入力が実数信号である場合、このパラメータは出力信号の一定の虚数部を指 定します。入力が虚数部である場合、このパラメータは出力信号の一定の実 数部を指定します。このフィールドのタイトルが、用法に応じて変わること に注意してください。

特性	直接フィールドスルー	あり
	サンプル時間	接続されるブロックから継承
	スカラ拡張	関数が2つの入力を必要とするときの入力が対象
	次元化	可
	ゼロクロッシング	なし

目的 指定した比較演算を入力に適用します。

ライブラリ Math

詳細

ぎの表に従って出力を生成します。

演算子	出力
==	最初の入力が2番目の入力と等しければTRUE
~=	最初の入力が2番目の入力と等しくなければTRUE
<	最初の入力が2番目の入力より小さければ TRUE
<=	最初の入力が2番目の入力以下であれば TRUE
>=	最初の入力が2番目の入力以上であれば TRUE
>	最初の入力が2番目の入力より大きければ TRUE

Relational Operator ブロックは、その2つの入力に対して比較演算を実行し、つ

結果が TRUE の場合の出力は 1 で、FALSE の場合の出力は 0 です。入力はスカ ラまたはベクトル、あるいはスカラとベクトルを組み合わせて指定することがで きます。

- スカラ入力の場合、出力はスカラです。
- ベクトル入力の場合、出力はベクトルです。この場合の各要素は、入力ベクトルの要素単位の比較の結果となります。
- スカラとベクトルが混在する入力の場合、出力はベクトルです。この場合の 各要素は、スカラと対応するベクトル要素との間の比較の結果となります。

ブロックアイコンは、選択した演算子を表示します。

サポートされて いるデータタイ プ Relational Operator ブロックは、任意のデータタイプの実数および複素数信号を 受け入れます。入力は双方共同じデータタイプでなければなりません。演算子が == または != である場合は、1 つの入力が実数でもう一方が複素数でもかまいま せん。ブロックは、boolean の整合性モードが保たれる限り(4-49 ページの"厳密 な Boolean 型の検査を有効にする"を参照)、boolean タイプの信号を出力します。 そして、ブロックは double の信号も出力します。

パラメータとダ イアログボック ス

Block Parameters: Relational Operator	×
C Relational Operator	
Relational operators.	
Parameters	-
Operator: <=	
OK Cancel <u>H</u> elp <u>Apply</u>	

Operator

ブロック入力に適用される比較演算子

特性

直接フィールドスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	入力に関して
次元化	可
ゼロクロッシング	あり、出力が変化する瞬間を検出するため

目的 2つの定数間で出力を切り替えます。

ライブラリ Nonlinear

詳細



Relay ブロックを用いると、2 つの指定した値の間で出力を切り替えることがで きます。リレーがオンの場合、入力が Switch off point パラメータの値以下にな るまでオンの状態です。リレーがオフの場合、入力が Switch on point パラメータ の値を越えるまでオフの状態です。ブロックは1つの入力を受け入れ、1 つの出 力を生成します。

Switch off point 値より Switch on point 値を大きく指定すると、ヒステリシスがモ デル化され、等しい値を指定するとその値をしきい値とするスイッチがモデル化 されます。Switch on point 値は Switch off point 値以上でなければなりません。

Relay ブロックは、double タイプの実数信号を受け入れ、出力します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Relay				
utput the specified becified thresholds etween the upper	d 'on' or 'off' value . The on/off stal and lower limits.	e by comparing te of the relay is	the input to the not affected by inpu	ut
Parameters				_
Switch on point:				
eps				
Switch off point:				
eps				
Output when on:				
Output when off:				
0				

Switch on point

リレーに対するオンのしきい値。デフォルトは eps。

Switch off point

リレーに対するオフのしきい値。デフォルトは eps。

Output when on

リレーがオンの場合の出力。デフォルトは1。

Output when off

リレーがオフの場合の出力。デフォルトは0。

特性

直接フィールドスルー	あり
------------	----

サンプル時間	接続されるブロックから継承
スカラ拡張	可
次元化	可
ゼロクロッシング	あり、スイッチがオンになる点とスイッチがオフにな る点を検出するため

目的 反復可能な任意の信号を発生します。

ライブラリ Sources

詳細



Repeating Sequence ブロックを用いると、時間の経過と共に規則的に反復される スカラ信号を出力することができます。任意の反復信号は、ブロックダイアログ の Time values、Output values パラメータで指定します。Times value パラメータ は、サンプル時間のベクトルを指定します。Output values パラメータは、対応す るサンプル時間に出力される信号の振幅のベクトルを指定します。2つのパラ メータは、信号が繰り返される(つまり信号の周期)区間の初めからの点によっ て、出力信号の1サンプルを指定します。たとえば、Time values と Output values パラメータは、デフォルトで両方とも[02]に設定されています。このデフォル ト設定により、シミュレーション開始から2秒毎に繰り返され、最大振幅が2の ギザギザの波形が設定されます。Repeating Sequence ブロックは、線形補間を 使って、指定したサンプル点とサンプル点の間の値を計算します。

このブロックは1次元のLook-Up Table ブロックを用いて実現され、ポイント間の線形補間を実行します。

Repeating Sequence ブロックは、double タイプの実数信号を出力します。

パラメータとダ イアログボック ス

サポートされて

いるデータタイ

プ

Block Parameters: Repeating Sequence
Repeating table (mask) (link)
Repeating table.
- Parameters
Time values:
[0 2]
Output values:
[0 2]
OK Cancel <u>H</u> elp <u>A</u> pply

Time values

単調増加する時間ベクトル。デフォルトは[02]。

Output values

出力値ベクトル。それぞれは同じ列の時間に対応します。デフォルトは [0 2]。

特性

サンプル時間	連続
スカラ拡張	不可
次元化	不可
ゼロクロッシング	なし

目的信号の大きさを変更します。

ライブラリ Signals & Systems

詳細

>U(:)>

Reshape ブロックは、入力信号の大きさを、ブロックの Output dimensionality パ ラメータを使って、指定した大きさに変更します。たとえば、このブロックを 使って、N- 要素のベクトルを 1xN または Nx1 行列信号に変更したり、その逆が 行えます。

Output dimensionality パラメータを使って、つぎの出力オプションの中から選択 することができます。

出力の大きさ Output Dimensionality	詳細
1 次元配列	行列 (2 次元配列) をベクトル (1 次元配列) 配列信号に変換します。出力ベクトルは、入力の1番目の列、その後に2番目の列、のように構成されます(このオプションは、ベクトル入力は変更しません)。
列ベクトル	ベクトルまたは行列入力信号を列行列、すなわち Mx1 行 列に変換します。ここで、M は入力信号の要素数です。 行列に対しては、変換は列単位の順番で行われます。
行ベクトル	ベクトルまたは行列入力信号を行行列、すなわち 1xN 行 列に変換します。ここで、N は入力信号の要素数です。 行列に対しては、変換は、列単位の順番で行われます。
カスタマイズ	入力信号を Output dimensions パラメータを使って指定し た大きさの出力信号に変換します。Output dimensions パ ラメータの値は、1 要素または 2 要素ベクトルです。値 [N] は、サイズ N のベクトルを出力します。値 [M N] は、 MxN 行列を出力します。入力信号の要素数は、Output dimensions パラメータで指定した要素数と一致しなけれ ばなりません。行列に対しては、変換は列単位の順番で 行われます。

サポートされて Reshape ブロックは、任意のタイプの信号を受け入れ、出力します。 いるデータタイ プ バラメータとダ Block Parameters: Reshape × イアログボック Reshape (mask) (link) Change the dimensions of a vector or matrix input signal. Dutput - a one-dimensional array (vector), - a column vector (Mx1 matrix), ス - a row vector (1xN matrix), or - a matrix or vector with specified dimensions, e.g., [M, N] or [W]. Parameters-Output dimensionality: 1-D array -[1,1] 0K Cancel <u>H</u>elp **Output dimensionality** 出力信号の大きさ

Output dimensions

カスタマイズした出力の大きさを指定します。このオプションは、Output dimensionality パラメータの値として Customize を選択した場合にのみ可能 です。

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	N/A
次元化	可
ゼロクロッシング	なし

目的 丸め関数を実行します。

ライブラリ Math

詳細

floor D

ユーザは、Function リストから、つぎの関数の1つを選択して実行することがで きます。Function リストに含まれる関数は、floor, ceil, round, fix です。ブロック の出力は、単数または複数の入力に関数を適用した結果です。Rounding Function ブロックは、double タイプの実数または複素数値信号を受け入れ、出力します。

Rounding Function ブロックは、一般的な数学丸め関数を実行します。

ブロックアイコンには、関数名が表示されます。

ベクトル化された出力を出力したい場合、Fcn ブロックではなく Rounding Function ブロックを使用してください。Fcn ブロックは、スカラ出力しか生成で きません。

サポートされて Rounding Function ブロックは、double タイプの実数信号を受け入れ、出力しま いるデータタイ す。

パラメータとダ イアログボック	Block Parameters: Rounding Function Rounding	×
ス	Parameters	

Function

丸め関数を選択します。

特性

プ

直接フィールドスルー あり サンプル時間 接続されるブロックから継承 スカラ拡張 N/A 次元化 可 ゼロクロッシング なし

Saturation

目的 信号の範囲を制限します。

ライブラリ Nonlinear

詳細

プ

Saturation ブロックは、信号の上限と下限を設定します。入力信号が Lower limit と Upper limit のパラメータで指定された範囲内にある場合、入力信号は変更さ れることなく通過します。入力信号がこれらの範囲外にある場合、信号は上限値 または下限値に固定されます。

パラメータを同じ値に設定すると、ブロックはその値を出力します。

サポートされて Saturation ブロックは、任意のデータタイプの実数信号を受け入れ、出力します。 いるデータタイ

パラメータとダ イアログボック ス

Block Parameters: S	aturation			į
Limit input signal to	the upper and low	er saturation v	alues.	
Parameters Upper limit: 0.5				
Lower limit: -0.5				
OK	Cancel	<u>H</u> elp	Apply	

Upper limit

入力信号に対する上限。信号がこの値より大きい場合、ブロック出力はこの 値に設定されます。

Lower limit

入力信号に対する下限。信号がこの値より小さい場合、ブロック出力はこの 値に設定されます。

特性

直接フィードスルー あり

- サンプル時間 接続されるブロックから継承
- スカラ拡張 パラメータと入力について

次元化 可

ゼロクロッシング あり、信号が限界に到達する瞬間と限界から離れる瞬 間を検出するため **目的** シミュレーション中に生成される信号を表示します。

ライブラリ Sinks

詳細

J	
1	

Scope ブロックは、シミュレーション時間に関して、その入力を表示します。 Scope ブロックは、複数の軸をもつことができます(端子に付き1つ)。すべて の軸は、y-軸に関してはお互い独立ですが、共通の時間レンジをもっています。 Scope を用いると、表示する時間や入力値の範囲を調整することができます。 Scope ウィンドウは、シミュレーション中に移動したり、サイズを変えたり、 Scope のパラメータ値を変更したりすることができます。

シミュレーションの開始時に、Simulink は、接続された Scope にデータを書き出 しますが、Scope ウィンドウは開きません。その結果、シミュレーション後に Scope を開くと Scope の入力信号 (1 つまたは複数)が表示されます。

信号が連続の場合、Scope ブロックは、点単位で連続的にプロット表示します。 信号が離散の場合、Scope ブロックは、階段状のプロット表示します。

Scope には、表示されるデータの拡大、Scope へのすべてのデータ入力の表示、 シミュレーション間での軸設定の保存、表示されるデータの制限、ワークスペー スへのデータの保存などを行うためのツールバーボタンがあります。ツールバー ボタンをつぎの図で示します。図は Scope ブロックを開いたときに表示される Scope ウィンドウを示しています。



注意 作成したライブラリブロック内部で Scope ブロックを使わないでください。 代わりに、ライブラリブロックに scope が内部データの表示用に接続することが できる出力端子を設定してください。

ベクトル信号の表示

ベクトル信号を表示する場合、Scope ブロックは、つぎの順序で異なる色を使用 します。すなわち、黄色、マゼンタ、シアン、赤、緑、ダークブルーの順です。 6つ以上の信号を表示する場合、Scope は上述の順番で色を繰り返し使用します。

Y 軸の範囲

軸をクリックして、Properties.... を選択することによって、 y 軸の範囲を設定す ることができます。このとき、つぎのようなダイアログボックスが表示されま す。

Scope' properties: axis 1		_ 🗆 X
Y-min: 3	Y-max: 3	
Title ('% <signallabel>' repla</signallabel>	aced by signal nam	ne):
% <signallabel></signallabel>		
OK	Cancel	Apply

Y-min

y 軸の最小値を設定します。

Y-max

y 軸の最大値を設定します。

Title

プロットのタイトルを設定します。タイトルの文字列に %<SignalLabel> (%<SignalLabel>を信号のラベルで置き換えてください)を加えることに よって、信号のラベルをタイトルに加えることもできます。

Time Offset

つぎの図は、vdp モデルの出力を表示した Scope ブロックを示しています。シ ミュレーションは 40 秒間実行されています。この Scope は、シミュレーション の最後の 20 秒を示していることに注意してください。Time offset フィールドは、 水平軸上の 0 に対応する時間を表示しています。そのため、x 軸上の固定された 時間範囲にオフセットを加えることによって実際の時間を算出することができる ようになります。



Scope 軸の自動スケーリング

つぎの図は、Auto-scale ツールバーボタンを押した後の同じ出力を示していますが、このボタンは両方の軸のスケールを自動的に調整して、保存されているすべてのシミュレーションデータを表示します。この例の場合、y軸はすでに適切な範囲に設定されているので、y軸に対するスケーリングは実施されません。



シミュレーションの実行中に Auto-scale ボタンをクリックすると、現在のスク リーン上に表示されているデータに基づいて軸が自動的にスケーリングされ、自 動スケールの範囲はデフォルトとして保存されます。これにより、他のシミュ レーションに対して同じ範囲を使用することができます。

データの拡大

データは、x軸とy軸の両方向同時に、あるいはいずれかの方向に別々に拡大することができます。シミュレーションの実行中は動作しません。

データを両方向同時に拡大する場合は、左端の Zoom ツールバーボタンが選択さ れていることを確認してください。それから、バウンディングボックスを使って ズーム領域を定義します。マウスボタンを解除すると、Scope は、その領域内の データを表示します。拡大したい部分の中の一点をクリックすることもできま す。

Scope が複数の y 軸を有している場合、その中の 1 組の x-y 軸を拡大すると、す べての x-y 軸は同一のタイムベース (x 軸)をもつ必要があるため、他の x-y 軸の 組全てについて、拡大した x-y 軸の組に適応する形で x 軸の範囲がそれぞれ変更 されることになります。



つぎの図は、図中に示すボックスの中に表示されているデータ部のみを表示しま す。

つぎの図は、マウスボタンをリリースした後、拡大された部分を示しています。



データをx方向にのみ拡大するには、Zoom ツールバーボタンをクリックします。ズーム領域は、ポインタを領域の一端に置き、マウスボタンを押したままの 状態でポインタを領域の他端に移動することにより定義します。つぎの図は、 ズーム領域を定義した後で、マウスボタンを解除する前の Scope を示しています。



マウスボタンを解除すると、Scopeは拡大された領域を表示します。拡大した部分内の一点をクリックしても拡大します。

y方向での拡大は、ズーム領域を定義する前に Zoom ツールバーボタンを押すことを除いて x 方向の場合と同じです。また、拡大した部分内の一点をクリックしても拡大します。

軸設定値の保存

Save axes settings ツールバーボタンを用いると、現在のx軸とy軸の設定値を保存し、それらをつぎのシミュレーションに適用することができます。



- Save axes settings ボタン

表示データの領域を拡大した後で軸設定値を保存し、別のシミュレーションで同じ領域を見ることもできます。時間範囲は、現在の *x* 軸範囲から推測します。

Scope のプロパティ

軸の数、軸の範囲、時間範囲、目盛りラベル、サンプリングパラメータの設定、 保存オプションの設定は、Properties ツールバーボタンを選択して変更できま す。

Properties ボタン

Properties ボタンをクリックすると、つぎのダイアログボックスが表示されます。

🜠 'Scope' properties 📃 🗌 🗙
General Data history Tip: try right clicking on axes
Axes
Number of axes: 1 Floating scope
Time range: auto
Tick labels: bottom axis only
Sampling
OK Cancel Help Apply

ダイアログボックスには General と Data history の 2 つのタブがあります。

General パラメータ

General タブで軸パラメータ、時間範囲、目盛りラベルを設定することができます。また、このタブから floating scope オプションを選択することもできます。

Number of axes

このデータフィールドでは、 y 軸の数を設定します。フローティングス コープを除けば、Scope ブロックが扱える軸の数に制限はありません。全て の軸が共通のタイムベース (x 軸)を共有することになりますが、y 軸につ いては独立の y 軸をもつことができます。軸の数が入力端子の数と等しいこ とに注意してください。

Time range

Time range フィールドに数値または auto を入力することにより、x 軸の範囲 を変更することができます。秒単位で数値を入力すると、指定された秒数に 対応する量のデータが各スクリーンに表示されます。一方、auto を入力する と、x 軸はシミュレーション時間に設定されます。このフィールドには変数 を入力しないでください。

Tick labels

全ての軸、または一つの軸、または Tick labels ドロップボックス内にのみある bottom axis 上に、目盛りラベルを割り当てるよう選択することができます。
Floating scope

フローティング Scope を使用するには、Floating scope チェックボックスを選 択してください。フローティング Scope は、1つまたは複数のラインに伝搬 される信号を表示することのできる Scope ブロックです。

フローティング Scope をモデルに追加するには、Scope ブロックをモデル ウィンドウにコピーして、ブロックを開きます。そしてブロックのツール バーの Properties ボタンを選択します。それから General タブを選択し、 Floating scope チェックボックスを選択します。

シミュレーション中にフローティング Scope を使用するには、まず Scope ブ ロックを開きます。ライン上を伝搬する信号を表示する場合はラインを選択 します。複数のラインを選択するには、Shift キーを押したまま別のライン をクリックします。信号によっては、信号が表示される軸を信号に合わせる ために、Scope ツールバー上の Auto-scale data ボタンを押す必要があります。 フローティングスコープは、複数の軸をもつことができないことに注意して ください。あるいは、表示する信号を選択するために、フローティング Scope の Signal Selector(9-217 ページの "Signal Selector"参照)が使用できま す。Signal Selector は、閉じたままのサブシステムも含めたモデルの任意の 場所の信号を選択することができます。

1 つのモデルを複数のフローティングスコープで表示することは可能です が、1スコープがアクティブにできるのは、1 軸だけです。アクティブなフ ローティングスコープは、アクティブな軸を青く表示します。ラインの選 択、非選択は、Scope ブロックにのみ影響を与えます。他のフローティング スコープは、信号がアクティブな状態だったときに、選択された信号を表示 し続けます。言い換えると、非アクティブなフローティングスコープは信号 が変化しない状態のままです。

シミュレーション中にフローティングスコープを使用したい場合、バッファ 再生機能を無効にしておかねばなりません。詳細は、5-31 ページの"信号ス トレージの利用"を参照してください。

Sampling

間引きファクタを指定するには、Decimationの選択肢の右側のデータフィー ルドに数字を入力します。サンプリング間隔でデータを表示するには、 Sample time の選択肢を選び、データフィールドに数字を入力します。

データ収集と表示の制御

Data History タブのフィールドを設定することにより、Scope ブロックにストアし、表示するデータ量を制御することができます。

🛃 'Scope' properties	
General Data history	Tip: try right clicking on axes
☑ Limit data points to last: 5	000
Save data to workspace	
Variable name: ScopeData	
Format: Structure with t	ime 🔽
OK Cano	el Help Apply

このタブでは、ワークスペースにデータを保存するオプションを選択できます。 Apply や OK ボタンをクリックしたときに、カレントのパラメータやオプション に適用されます。入力フィールドに表示されている値は、つぎのシミュレーショ ン実行時に使用されます。

Limit data points to last

Limit rows to last チェックボックスをチェックし、データフィールドに値を 入力することにより、ワークスペースに保存されるデータ点の数を制限する ことができます。Scope は、拡大や自動スケーリングを行う際にデータの履 歴を利用します。行の数が 1000 に制限されている時に、シミュレーション によって 2000 の行が生成された場合、最後の 1000 行だけが表示の再生に利 用されることになります。

Save data to workspace

Save data to workspace チェックボックスをチェックすることにより、Scope が収集したデータはシミュレーション終了時に自動的に保存されます。この オプションをチェックすると、Variable name と Format フィールドが共にア クティブとなります。

Variable name

Variable name フィールドに変数の名前を入力してください。指定した名前 は、モデル内で使用される全てのデータログ変数の中で一意的でなければな りません。他のデータログ変数は、Scope ブロック、To Workspace ブロッ ク、及び時間、状態、出力などのシミュレーションの出力変数で定義されま す。Scope データをワークスペース上に保存できるので、同じデータスト リームを Scope ブロックと To Workspace ブロックの両方に送る必要があり ません。

Format

データは、つぎの3つのフォーマットのいずれか1つを用いて保存すること ができます。利用できるフォーマットは、Matrix, Structure, Structure with time です。1つの軸をもつ Scope ブロックに対してのみ、Matrix を使用してくだ さい。複数の軸をもつ Scope については、時間データを保存する必要がなけ れば Structure を使い、また時間データを保存する必要があれば、Structure with time を使ってください。

Scope ウィンドウの内容を印刷する

Scope ウィンドウの内容を印刷するには、Scope ツールバーの最も右側にある Print アイコンをクリックして Scope Properties ダイアログを開いてください。

Print icon

サポートされて Scope ブロックは、すべての要素のデータタイプが同じであるベクトルも含め、 いるデータタイ 任意のタイプの実数信号を受け入れます。

プ

特性 サンプル時間 接続されるブロックから継承 状態 0 目的 入力要素を選択します。

ライブラリ Signals & Systems

詳細

> Selector Selector ブロックは、入力ベクトルの要素を選択して出力を生成します。

Slector ブロックは、入力としてベクトルまたは行列信号を受け入れます。Input Type パラメータは、ブロックが受け入れる信号のタイプ(ベクトルまたは行列) に設定してください。パラメータダイアログボックスとブロックアイコンは、 ユーザが選択した入力のタイプによって異なります。ブロックが選択する要素を 決定する方法は、入力のタイプに依って、すこしづつ異なります。

ベクトル入力

入力タイプがベクトルの場合、Selector ブロックは、選択要素のベクトルを出力 します。またプロックは、Elements パラメータ、または外部信号から選択する 要素のインデックスを決定します。Source of element indices パラメータをソース (internal つまりパラメータ値、または external) に設定します。もし、external を 選択すると、ブロックは外部インデックス信号のための入力端子を加えます。

どちらの場合でも、選択された要素が1要素でない限り、ベクトルにしなければ なりません。たとえば、つぎのモデルは、Selector ブロックのアイコンと、[246 810]の入力ベクトルのための出力と、Elements パラメータの値が[513]を示し ています。



ブロックアイコンは、入力ベクトル要素の順序をグラフィカルに表示します。ブ ロックの大きさが十分でない場合は、ブロック名を表示します。

要素インデックスのソースに external を選択した場合、要素インデックス信号の 入力端子がブロックに加えられます。信号は、Elements パラメータを使って指 定したのと同じ方法で選択された要素を指定してください。

入力タイプがベクトルの場合、Input port width パラメータを使って、入力信号の幅を与えるか、-1を指定してください。0より大きな幅を指定する場合、入力 信号の幅は、指定した幅と等しくなるようにしてください。そうでなければ、ブ ロックはエラーをレポートします。-1 の幅を指定した場合、ブロックは任意の 幅のベクトル信号を受け入れます。

行列入力

入力タイプが行列の場合、Selector ブロックは入力行列から選択された要素の行列を出力します。ブロックは、Rows と Columns パラメータ、または外部信号から要素を選択するために、行と列のインデックスを決めます。ブロックのSource of row indices と Source of column indices を選択したい (internal あるいは external) ソースに設定してください。external ソースを設定した場合、ブロックは外部インデックス信号のために入力端子を付け加えます。両方とも external を選択すると、2 つの入力端子が付け加えられます。

どちらの場合でも、選択する行と列のインデックスは、ベクトル(あるいは1行 1列の場合のみスカラ)で指定しなければなりません。たとえば、Rowsが[21] で Columns が[13]の場合、2×2の行列の出力を指定します。このとき、出力 は、最初の行が入力行列の2番目の行の1番目と3番目の要素を、そしてその次 の行が入力行列の最初の行の1番目と3番目の要素を含みます。



 サポートされて
 Selector ブロックは、様々なタイプの信号ベクトルが混ざっているものも含めた

 いるデータタイ
 任意の信号とデータタイプを受け入れます。出力ベクトルの要素は、対応する選

 プ
 扱された入力要素と同じタイプです。

パラメータとダ ベクトル入力モードを選択すると、つぎのパラメータダイアログボックスが表示 **イアログボック** されます。

ス

Block Parameters: Selector	×
Selector-	
Select or re-order the specified elements of the input vector. y=u(Elements).	
Parameters	
Input Type: Vector	
Source of element indices:	
Elements (-1 for all elements):	
[1 3]	
Input port width:	
3	
OK Cancel <u>H</u> elp <u>Apply</u>	

Input Type

入力信号のタイプ:ベクトルあるいは行列

Source of element indices

要素を選択して指定するインデックスのソース。internal つまり Elements パ ラメータか、あるいは external つまり入力信号のどちらかです。

Elements

出力ベクトルに含まれる要素

Input port width

入力ベクトルの要素の数

行列入力モードを選択すると、つぎのダイアログボックスが表示されます。

Block Parameters: Selector			
Selector			
Select or re-order the specified elements of the input vector. y=u(Elements).			
Parameters			
Input Type: Matrix			
Source of row indices: Internal			
Rows (-1 for all rows):			
1			
Source of column indices: Internal			
Columns (-1 for all columns):			
1			
OK Cancel <u>H</u> elp <u>A</u> pply			

Input Type

入力信号のタイプ:ベクトルまたは行列

Source of row indices

入力行列から選択する行を指定するインデックスのソース。internal つまり Rows パラメータか、あるいは external つまり入力信号のどちらかです。

Rows

出力行列に含まれる要素を選択するための行のインデックス。

Source of column indices

入力行列から選択する列を指定するインデックスソース。internal つまり Columns パラメータか、あるいは external つまり入力信号のどちらかです。

Columns

出力行列に含まれる要素を選択するための列のインデックス。

サンプル時間	接続されるブロックから継承
次元化	可

目的 S-ファンクションにアクセスします。

ライブラリ Functions & Tables

詳細

> system

S-Function ブロックは、ブロック線図から S- ファンクションへのアクセスを提供します。S-function name パラメータで指定できる S- ファンクションは、S-ファンクションとして書かれた M- ファイルまたは MEX- ファイルです。

S-Function ブロックでは、追加パラメータを指定の S- ファンクションに直接渡 すことができます。関数パラメータは、MATLAB 表現またはコンマで区切られ た変数として指定することができます。たとえば、つぎの通りです。

A, B, C, D, [eye(2,2);zeros(2,2)]

個々のパラメータは鍵括弧で囲むことができますが、パラメータのリストは鍵括 弧で囲んではいけません。

S-Function ブロックは、指定した S-ファンクション名を表示し、含まれるサブシステムの入力数と出力数に関係なく、常に1つの入力端子と1つの出力端子を伴って描かれます。

S-ファンクションに複数の入力や出力が含まれる場合は、ベクトルラインが使用されます。入力ベクトルの幅は、S-ファンクションに含まれる入力数と一致しなければなりません。プロックは入力ベクトルの最初の要素を S-ファンクションの最初の入力へ、2番目の要素を2番目の入力へという具合に対応させます。同様に、出力ベクトルの幅は、S-ファンクションの出力数と一致しなければなりません。

サポートされて いるデータタイ プ S-Function ブロックの実現に依存

パラメータとダ イアログボック ス

特性

Block Parameters: S-F	unction			×
- S-Function				
User-definable block. Blocks may be written in M, C or Fortran and must conform to S-function standards. tx,u and flag are automatically passed to the S-function by Simulink. "Extra" parameters may be specified in the "S-function parameters' field.				
Parameters				
S-function name:				
system				
S-function parameters:				
OK	Cancel	<u>H</u> elp	Apply	

S-function name

S-function 名

S-function parameters

追加の S-function パラメータ。パラメータの指定法については、前のブロックの説明を参照してください。

直接フィードスルー	S-function の内容に依存
サンプル時間	S-function の内容に依存
スカラ拡張	S-function の内容に依存
次元化	S-function の内容に依存
ゼロクロッシング	なし

目的 入力の符号を示します。

Math

ライブラリ

詳細

Sign ブロックは入力の符号を示します。

- 入力がゼロより大きい場合、出力は1となります。
- 入力がゼロの場合、出力は0となります。
- 入力がゼロより小さい場合、出力は-1となります。

double タイプの実数信号を受け入れ、出力します。

サポートされて いるデータタイ プ

Dialog Box

Bloc	k Parameters: Sij	gn			×
– Sig Out sign	gnum tput 1 for positive num(u)	e input, -1 for nega	ative input, and 0	for 0 input. y =	
	OK	Cancel	<u>H</u> elp	Apply	

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	N/A
次元化	可
ゼロクロッシング	あり、入力がゼロを横切る時点を検出するため

_ 🗆 ×

目的 種々の波形を発生します。

ライブラリ Sources

詳細



Signal Generator ブロックは、正弦波、矩形波、ノコギリ波の3種類の波形から1 つを発生することができます。信号パラメータは、ヘルツ(デフォルト)または ラジアン / 秒単位で表示することができます。つぎの図は、デフォルトパラメー タ値を用いて Scope 上に表示されたそれぞれの信号を示しています。



Sine Wave

Square Wave



Sawtooth Wave

負の Amplitude パラメータ値は、位相が 180 度シフトされます。180 度以外で、 位相をシフトした波形を発生させるには、Clock ブロック信号を MATLAB Fcn ブロックに入力して特定の波形に対する方程式を作成するなどのいくつかの方法 があります。

シミュレーションの実行中、Signal Generator ブロックの出力設定値を変更することができます。これは、種々の入力に対するシステムの応答をすばやく決定するのに有効です。

ブロックの Amplitude と Frequency パラメータは、出力信号の振幅と周波数を決めます。パラメータは、スカラ拡張後同じ次元にならなければなりません。 Interpret vector parameters as 1-D オプションがオフの場合、ブロックは、(スカラ 拡張後) Amplitude と Frequency パラメータと同じ次元の信号を出力します。 Interpret vector parameters as 1-D オプションがオンの場合、ブロックは Amplitude と Frequency パラメータが行あるいは列ベクトル、つまり単一行あるいは単一列 の 2 次元配列ならば、1 ベクトル(1 次元)信号を出力します。 ブロックはパラメータと同じ次元の信号を出力します。

Signal Generator ブロックは、double タイプの実数信号を出力します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Signal Generator	×
- Signal Generator	
Output various wave forms.	
Parameters	
Wave form: Sine	
Amplitude:	
1	
Frequency:	
1	
Units: Hertz	
☑ Interpret vector parameters as 1-D	
OK Cancel Help Apply	

Wave form

波形:正弦波、矩形波、ノコギリ波。デフォルトは正弦波。

Amplitude

信号の振幅。デフォルトは1。

Frequency

信号の周波数。デフォルトは1。

Units

信号の単位。ヘルツまたはラジアン/秒。デフォルトはヘルツ。

Interpret vector parameters as 1-D

選択すると、Amplitude と Frequency パラメータの行または列行列の値は、ベクトル信号となり出力されます。

サンプル時間	連続
スカラ拡張	パラメータについて
次元化	可
ゼロクロッシング	なし

目的 正弦波を発生します。

ライブラリ Sources

詳細

Sine Wave ブロックは正弦波を発生します。ブロックは、連続モードまたは離散 モードのいずれでも動作することができます。

Sine Wave ブロックの出力は、以下のようになります。

 $y = Amplitude \times sin(frequency \times time + phase)$

Sample time パラメータの値は、ブロックが連続モードまたは離散モードのどち らで動作するのかを決定します。

- 0(デフォルト)の場合、ブロックは連続モードで動作します。
- >0の場合、ブロックは離散モードで動作します。
- -1の場合、ブロックは信号を受け取るブロックと同じモードで動作します。

Sine Wave ブロックの離散モードでの使用

Sample time パラメータ値がゼロより大きいと、ブロックは、サンプル時間をその値に設定した Zero-Order Hold ブロックを接続しているのと同様の挙動を示します。

Sine Wave ブロックをこの方法で用いると、ハイブリッド連続/離散システムの モデルではなく、純粋に離散的である正弦波入力をもつモデルを構築することが できます。ハイブリッドシステムは、本質的により複雑で、結果としてシミュ レーションに時間を必要とします。

離散モードの Sine Wave ブロックは、絶対時間に基づくアルゴリズムではなく、 増分アルゴリズムを使用します。その結果、ブロックは振動検査や疲労検査な ど、不定時間長で実行されるモデルで有効です。

増分アルゴリズムは、前のサンプル時間で計算した値に基づいて正弦値を計算し ます。この手法はつぎの恒等式を使用します。

 $\sin(t + \Delta t) = \sin(t)\cos(\Delta t) + \sin(\Delta t)\cos(t)$

 $\cos(t + \Delta t) = \cos(t)\cos(\Delta t) - \sin(t)\sin(\Delta t)$

これらの恒等式は、つぎのような行列形式で表わすことができます。

$$\begin{bmatrix} \sin(t + \Delta t) \\ \cos(t + \Delta t) \end{bmatrix} = \begin{bmatrix} \cos(\Delta t) & \sin(\Delta t) \\ -\sin(\Delta t) & \cos(\Delta t) \end{bmatrix} \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix}$$

∆t は一定なので、つぎの式は定数となります。

 $\begin{bmatrix} \cos(\Delta t) & \sin(\Delta t) \\ -\sin(\Delta t) & \cos(\Delta t) \end{bmatrix}$

したがって、これは sin(t) の値に定数行列を乗算して sin(t+t) を求める、行列乗算 問題の 1 つとなります。

Sine Wave ブロックの連続モードでの使用

Sample time パラメータ値がゼロであると、ブロックは連続モードでの挙動を示します。連続モードで動作している場合、時間が非常に大きくなるに従って、 Sine Wave プロックは、精度の低下により、不正確になる可能性があります。

ブロックの数値パラメータはスカラ拡張後同じ次元にならなければなりません。 Interpret vector parameters as 1-D オプションがオフの場合、ブロックはパラメータ と同じ次元の信号と次元を出力ます。Interpret vector parameters as 1-D オプショ ンがオンで、数値パラメータが行または行ベクトル(つまり単一行または列の2 次元配列)の場合、ブロックは1ベクトル(1次元配列)信号を出力します。そう でない場合、ブロックはパラメータと同じ次元の信号と次元を出力します。

Sine Wave ブロックは、double タイプの実数信号を受け入れ、出力します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Sine Wave		
Sine Wave		
Output a sine wave.		
Parameters		
Amplitude:		
1		
Frequency (rad/sec):		
1		
Phase (rad):		
0		
Sample time:		
0		
☑ Interpret vector parameters as 1-D		
OK Cancel Help Apply		

Amplitude

信号の振幅。デフォルトは1。

Frequency

ラジアン / 秒単位の周波数。デフォルトは1 ラジアン / 秒。

Phase

ラジアン単位での位相シフト。デフォルトは0ラジアン。

Sample time

サンプル時間。デフォルトは 0。

Interpret vector parameters as 1-D

選択すると、Sine Wave ブロックの数値パラメータのための列または行の行列の 値がベクトルの出力信号になります。そうでなければ、ブロックはパラメータと 同じ次元の信号を出力します。このオプションが選択されなければ、ブロックは ブロックの数値パラメータと同じ次元の信号と常に出力します。

サンプル時間	連続または離散または継承
スカラ拡張	パラメータについて
次元化	可
ゼロクロッシング	なし

目的 スライダを使って、スカラゲインを変化させます。

ライブラリ Math

詳細 Slider Gain ブロックでは、スライダを使って、シミュレーション中にスカラゲイ ンを変化させることができます。ブロックは1つの入力を受け入れ、1つの出力 を生成します。

サポートされて Slider Gain ブロックがサポートするデータタイプは、Gain ブロックがサポートすいるデータタイプ るデータタイプと同じです (9-108 ページの "Gain" を参照)。

ダイアログボッ クス

プ

🛃 Slider Gain		
•		•
Low		High
0	1	2
	Help	Close

Low

スライダ範囲の下限。デフォルトは0。

High

スライダ範囲の上限。デフォルトは2。

エディットフィールドは(左から右に)下限、現在値、上限を示しています。ス ライダを操作するか、新しい値を現在値のフィールドに入力するかの2つの方法 で、ゲインを変更することができます。ゲインの値の範囲は、下限と上限を変更 することによって変えられます。Close ボタンをクリックしてダイアログボック スを閉じてください。

スライダの左矢印または右矢印をクリックすると、現在の値はスライダの範囲の 約1%変化します。スライダのインジケータのいずれかの側の長方形領域をク リックすると、現在の値はスライダの範囲の約10%変化します。

ブロック入力にベクトルゲインを適用するには、Gain ブロックを使用してください。

特性

直接フィードスルーありサンプル時間接続されるブロックから継承スカラ拡張ゲインについて状態0次元化可ゼロクロッシングなし

目的 線形状態空間システムを実現します。

ライブラリ Continuous

詳細

State-Space ブロックは、以下によって挙動が定義されるシステムを実現します。

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

ここで、xは状態ベクトル、uは入力ベクトル、yは出力ベクトルです。行列係数は、下の図に示すように、つぎのような特性をもっていなければなりません。

- A は、n 行 n 列の行列でなければなりません。n は状態数です。
- Bは、n行m列の行列でなければなりません。mは入力数です。
- Cは、r行n列の行列でなければなりません。rは出力数です。
- Dは、r行m列の行列でなければなりません。



ブロックは、1 つの入力を受け入れ、1 つの出力を生成します。入力ベクトルの 幅は、B 行列とD 行列の列数で決まります。出力ベクトルの幅は、C 行列とD 行列の行数で決まります。

Simulink は、ゼロを含む行列をスパース行列に変換して、乗算の効率を高めます。

A State-Space ブロックは、double タイプの実数信号を受け入れ、出力します。

サポートされて いるデータタイ プ

State-Space

パラメータとダ イアログボック ス



A, B, C, D

行列係数

Initial conditions

初期状態ベクトル

直接フィードスルー	D≠0の場合のみ
サンプル時間	連続
スカラ拡張	初期条件について
状態	A のサイズに依存
次元化	可
ゼロクロッシング	なし

目的 Step 関数を発生します。

ライブラリ Sources

詳細



Step ブロックは、指定時間で2つの定義可能なレベル間のステップを提供しま す。シミュレーション時間が Step time パラメータ値より小さい場合、ブロック の出力は Initial value パラメータ値となります。シミュレーション時間が Step time 以上の場合、出力は Final value パラメータ値となります。

ブロックの数値パラメータは、スカラ拡張後同じ次元になるようにしなければな りません。Interpret vector parameters as 1-D オプションがオフの場合、ブロック はパラメータと同じ次元の信号と次元を出力します。Interpret vector parameters as 1-D オプションがオンで、数値パラメータが行または列ベクトル(つまり単一 行または列の2次元配列)の場合、ブロックは1ベクトル(1次元配列)信号を出 力します。そうでなければ、ブロックはパラメータと同じ次元の信号と次元を出 力します。

サポートされて いるデータタイ プ Step ブロックは、double タイプの実数信号を出力します。

パラメータとダ イアログボック ス

Step				
Output a step.				
Parameters				
Step time:				
1				
Initial value:				
0				
Final value:				
1				
Sample time:				
0				
☑ Interpret vector parameters as 1-D				
OK	Cancel	<u>H</u> elp	Apply	1

Step time

出力が Initial value パラメータから Final value パラメータにジャンプする、 秒単位の時間。デフォルトは 1 秒。 特性

Initial value

シミュレーション時間が Step time パラメータに到達するまでのブロック出力。デフォルトは 0。

Final value

シミュレーション時間が Step time パラメータに到達してそれを越えたとき のブロック出力。デフォルトは 1。

サンプル時間

ステップのサンプルレート

Interpret vector parameters as 1-D

選択すると、Step ブロックの数値パラメータの列または行の行列の値が1ベクトル出力信号となります。そうでなければ、ブロックはパラメータと同じ次元の 信号を出力します。このオプションを選択しない場合、ブロックは、常に、ブロックの数値パラメータと同じ次元の信号を出力します。

接続されるブロックから継承
パラメータについて
可
あり、ステップ時間を検出するため

目的 入力が非ゼロの場合にシミュレーションを停止します。

ライブラリ Sinks

詳細



Stop Simulation ブロックは、入力が非ゼロの場合にシミュレーションを停止しま す。

シミュレーションは、終了前に現在の時間ステップを完了します。ブロック入力 がベクトルの場合は、いずれかのベクトル要素が非ゼロであれば、シミュレー ションを停止します。

このブロックを Relational Operator ブロックと組み合わせて使うことにより、シ ミュレーションの停止時間を制御することができます。たとえば、つぎのモデル では、入力信号が10に到達すると、シミュレーションが停止します。

Stop Simulation ブロックは、double と boolean タイプの実数信号を受け入れます。



サポートされて いるデータタイ プ

ダイアログボッ クス	Block Parameters: Stop Simulation Stop Stop simulation when input is non-zer	io.
特性	サンプル時間	接続されるブロックから継承
	次元化	可

詳細

目的 システムをまとめて、別の1つのシステムで表現します。

ライブラリ Signals & Systems

Subsystem ブロックは、あるシステムをまとめて、別の1つのシステムで表現し ます。サブシステムは、つぎの方法で作成します。

- Subsystem ブロックを Signals & Systems ライブラリからモデル内にコピーします。それから Subsystem ブロックを開いて、必要なブロックをそのウィンドウにコピーすることにより、ブロックをサブシステムに追加することができます。
- バウンディングボックス(マウスボタンを押したまま枠をとったもの)を用いて サブシステムを構成するブロックとラインを選択した後、Edit メニューから Create Subsystem を選択します。Simulink は、ブロックを Subsystem ブロック に置き換えます。ブロックを開くと、ウィンドウは選択したブロックを表示 します。Inport ブロックと Outport ブロックをサブシステムへの入出力信号を 参照する位置に加えます。

Subsystem ブロックのアイコン上に描画される入力端子の数は、サブシステム内の Inport ブロックの数に対応します。同様に、ブロック上に描画される出力端子の数は、サブシステム内の Outport ブロックの数に対応します。

サブシステムに関する詳細は第3章の "Subsystems の作成"を参照してください。

パラメータとダ イアログボック ス

Show port labels		
Read/Write permissions:	ReadWrite	▼
Name of error callback fu	nction:	
🗖 (Treat as atomic unit		
RTW system code: Fu	nction	7
RTW function name opti	ons: UserSpecified	7
RTW function name:		
RTW file name opt <u>ions:</u>	Auto	~
BTW file name (no exten	sioni:	

Show port labels

サブシステムアイコン内にサブシステムの端子を表示します。

Treat as atomic unit

Simulink は、ブロックの実行順序を決定する時に、サブシステムを1ユニットとして取り扱います。サブシステムを実行するのに時間がかかる場合、 Simulink はサブシステムブロックと同じレベルに存在する他のブロックを実行する前に、サブシステム内のすべてのブロックを実行します。このオプションを選択しない場合、ブロックの実行順序を決定する時、Simulink は、 サブシステム内のすべてのブロックをサブシステムと同様に、モデルの階層 内の同じレベルにあるものとして扱います。これによって、サプシステム内 のブロックの実行は、サブシステム外のブロックの実行と交互に行われま す。詳細は、3-13 ページの "Atomic サブシステムと仮想サブシステム"を参 照してください。

Access

サブシステムのユーザ権限を制御します。次の値から選択することができま す。

権限	説明
ReadWrite	ユーザは、サブシステムを開いたり内容を修正する ことができます。
ReadOnly	ユーザは、サブシステムを開くことはできますが、 修正するこはできません。サブシステムがブロック ライブラリに属している場合、ユーザはサブシステ ムへのリンクを作成したり開いたりすることはでき、 サブシステムのローカルコピーを修正することがで きますが、オリジナルライブラリの内容を修正した リパーミションを変更することはできません。
NoReadOrWrite	ユーザはサブシステムを開いたり修正することがで きません。サブシステムがライブラリに属する場合、 ユーザは、モデル内にサブシステムへのリンクを作 成することはできますが、開いたり修正したり、 パーミションを変更したり、サブシステムのローカ ルコピーを作成することはできません。

Name of error callback function

サブシステムの実行中に生じたエラーを呼び出す関数の名前。Simulink は、 関数に2つの引数を渡します。サブシステムのハンドルとエラータイプを指 定する文字列です。関数が指定されなかった場合、Simulink は、サブシステ ムの実行中にエラーが生じた場合、一般的なエラーメッセージを表示しま す。

注意 RTW で始まるパラメータの名前はコード生成に必要な Real-Time Workshop が使用しています。詳細は、Real-Time Workshop のマニュアルをご覧ください。

特性

次元化

- サンプル時間 サブシステム内のブロックに依存
 - サブシステム内のブロックに依存
- ゼロクロッシング あり、イネーブル端子とトリガ端子が存在すれば、それに関して

目的 入力の総和を計算します。

ライブラリ Math

詳細



Sum ブロックは、スカラ入力、ベクトル入力、行列入力、またはベクトル入力の要素を加算します。つぎの規則によってブロックの出力を決定します。

- ブロックに複数の入力がある場合、すべての非スカラ入力は同じ大きさでなければなりません。つまり、すべて同じ大きさのベクトルまたは行列でなければなりません。たとえば、入力が2行2列行列である場合、それ以外の入力は2行2列の行列またはスカラでなければなりません。
- 入力がスカラである場合、非スカラ入力と同じ大きさになるように拡張されます。たとえば、非スカラ入力が2行2列の行列である場合、スカラ入力は2行2列の行列に拡張されます。
- 出力は、(スカラ拡張後の)入力と同じ大きさをもち、各要素は入力の対応する 要素の和です。言い換えると、出力は入力の要素単位の和です。
- ブロックが1入力のみをもつ場合、それはスカラまたはベクトルでなければなりません。入力がベクトルの場合、出力は入力ベクトルの要素の和と等しいスカラです。

注意 Simulink は、Sum ブロックを Simulink ブロックライブラリからモデルにコ ピーする場合は、その名前を非表示にします。

サポートされて Sum ブロックは、任意のデータタイプの実数値または複素数値信号を受け入れ **いるデータタイ** ます。全ての入力は、同じデータタイプでなければなりません。出力データタイ プ プは、入力データと同じデータタイプになります。

パラメータとダ イアログボック ス

Block Parameters: Sum	×
Sum Add or subtract inputs. Specify one of the following: a) string containing + or - for each input port, for spacer between ports (e.g. ++ ++) b) scalar >= 1. A value > 1 sums all inputs; 1 sums elements of a single input vector	
Parameters I con shape: tound List of signs:	
I++ ✓ Saturate on integer overflow	
OK Cancel <u>H</u> elp Apply	

Icon shape

Icon shape ドロップボックスを使って、Sum ブロックを円または長方形のいずれで表示するかを選択することができます。Sum ブロックが複数の入力をもっている場合、長方形よりも円で表示するほうが便利です。

List of signs

List of signs パラメータには、定数または+符号と-符号、そして、 | 記号の 組み合わせを入力できます。定数を指定すると、Simulink は指定された数の 端子をもつブロックを全て正の極性で再描画します。+と-の符号を組み合 わせると、各端子の極性がそれによって決定され、端子数は使用する符号の 数に等しくなります。

Sum ブロックは、端子の横に + 符号や - 符号を表示し、List of signs パラメー タに設定している符号の数に一致する端子を再描画します。符号の数を変更 すると、端子はアイコンに加えられたり、削除されたりします。必要なら ば、Simulink は、すべての入力端子を表わすようにブロックをリサイズしま す。List of signs パラメータの中に | 記号で区切った符号を設定することで入 力端子の位置を決定することができます。 | 記号は、端子間に余分なスペー スを設定します。たとえば、++|-- は、2番目の + 端子と最初の - 端子の間に スペースを設定します。

Saturate on integer overflow

このオプションを選択すると、Sum ブロックの出力が整数オーバフローの原 因になります。特に、出力データタイプが整数タイプの場合、出力タイプに より表現される最大値かまたは計算された出力に関して、絶対値の意味で小 さいほうの値をブロック出力とします。オプションを選択しない場合、 Simulink は、Simulation Parameters ダイアログの Diagnostics ページ上の **Data overflow** で設定された挙動を示します。(詳細は、5-25 ページの "Diagnostics ページ"を参照してください)。

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	可
状態	0
次元化	可
ゼロクロッシング	なし

目的 2つの入力の切り替えを行います。

ライブラリ Nonlinear

詳細

プ

ス

パラメータとダ イアログボック



Switch ブロックは、制御入力と呼ばれる第3の入力に応じて、2つの入力の1つ を出力に伝搬します。制御(2番目)入力上の信号が Threshold パラメータ以上の 場合、ブロックは最初の入力を伝搬し、そうでない場合は3番目の入力を伝搬し ます。つぎの図は、このブロック端子の使い方を示したものです。



論理入力(すなわち、0か1)で切り替えを行うためには、しきい値を0.5に設定 します。

サポートされて Switch ブロックの切り替え対象となる入力(入力1と3)には、任意のデータタ いるデータタイ イプの実数または複素数値の信号を入力することができます。ただし、切り替え 対象となる入力は、双方、同一のタイプでなくてはなりません。Switch ブロッ クの出力信号は、選択された入力のデータタイプと同じになります。一方、 Threshold 入力は、bool または double のタイプでなければなりません。

Switch	IS. OWNER	1				
Pass through i otherwise, pas	nput 1 wi s through	hen input 2 n input 3.	is greal	er than c	r equal	to threshold;
- Parameters -						
Threshold:						
0						

Threshold

スイッチを他の状態に切り替える制御(2番目の入力)の値。このパラメータ は、スカラまたは入力ベクトルと等しい幅のベクトルとして指定することが できます。

直接フィードスルー あり サンプル時間 接続されるブロックから継承

スカラ拡張	可
次元化	可
ゼロクロッシング	あり、スイッチ条件が発生する瞬間を検出するため

目的 未接続の出力端子を終端させます。

ライブラリ Signals & Systems

プ

 詳細 Terminator ブロックは、出力端子が他のブロックに接続されていないブロックを 終端させるのに使用することができます。未接続の出力端子をもつブロックを用 いてシミュレーションを実行すると、Simulink はワーニングメッセージを表示し ます。Terminator ブロックを使ってこれらのブロックを終端させると、ワーニン グメッセージを回避することができます。

サポートされて Terminator ブロックは、任意の数値タイプまたはデータタイプの信号を受け入れ **いるデータタイ** ます。

パラメータとダ イアログボック ス	Block Parameters: Terminator Terminator Used to "terminate" output signals. (Preve unconnected output ports.) OK Cancel	nts warnings about
特性	サンプル時間 次元化	接続されるブロックから継承 可

目的

データをファイルに書き出します。

ライブラリ Sinks

詳細

ontitled.mat

To File ブロックは、その入力を MAT-ファイル内の行列に書き出します。ブロックは各時間ステップごとに1つの列を書き出します。最初の行はシミュレーション時間です。各列の残りの行は入力データで、入力ベクトルの各要素に対して1つのデータ点が割り当てられます。行列は、つぎのような形式になります。

$$\begin{bmatrix} t_1 & t_2 & \dots & t_{final} \\ u & 1_1 & u & 1_2 & \dots & u \\ \dots & & & \\ u & n_1 & u & n_2 & \dots & u & n_{final} \end{bmatrix}$$

From File ブロックは、To File ブロックが作成したデータを修正しないで使用することができます。しかし、From Workspace ブロックが期待する行列の形式は、To File ブロックで作成したデータの転置行列です。

このブロックは、シミュレーション完了後にデータとシミュレーション時間を書 き出します。ブロックアイコンは、指定した出力ファイル名を表示します。

書き出されるデータの量とデータが書き出される時間ステップは、ブロックパラ メータによって決まります。

- Decimation パラメータを用いると、n番目のサンプルごとにデータを書き出す ことができます。nは間引きファクタです。デフォルトの間引きは1で、すべ ての時間ステップ毎にデータを書き出します。
- Sample time パラメータでは、データ点を収集するサンプリング間隔を指定することができます。このパラメータは、時間ステップの間隔が同じでない可変ステップソルバを使用する場合に有効です。デフォルト値は-1 で、その場合は、どの点を書き出すかを決定する際に、接続ブロックからサンプル時間を継承します。

シミュレーションを開始するときに指定のファイルが存在すると、ブロックはその内容を上書きします。

サポートされて To File ブロックは、double タイプの実数信号を受け入れます。

いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: To File 🛛 💌
To File
Write time and input to specified MAT file in row format. Time is in row 1.
Parameters
Filename:
untitled.mat
Variable name:
ans
Decimation:
1
Sample Time:
-1
OK Cancel <u>H</u> elp <u>Apply</u>

Filename

行列を保持する MAT-ファイル名

Variable name

指定のファイルに含まれる行列名

Decimation

間引きファクタ。デフォルト値は1。

Sample time

データ点を収集するサンプル時間

可

接続されるブロックから継承

特性

サンプル時間 次元化 目的 データをワークスペース内に書き出します。

ライブラリ

詳細

simout

To Workspace ブロックは、入力データをワークスペースに書き出します。ブロックは、その出力を Variable name パラメータで設定した名前をもつ行列または構造体に、その出力を書き出します。Save format パラメータは、出力フォーマットを決定します。

Array

Sinks

このオプションを選択すると、To Workspace ブロックには N-次元配列で入力が 保存されます。ここで、N は入力信号の次元の数よりも1つ大きい次元です。た とえば、入力信号が1次元配列(つまりベクトル)の場合、ワークスペース配列 は2次元になります。入力信号が2次元配列(つまり行列)の場合は、配列は3 次元になります。

サンプルが配列に保存される方法は、入力信号がスカラかベクトルか行列かに依 り異なります。入力がスカラまたはベクトルの場合、各々の入力サンプルは、配 列の行として出力されます。たとえば、出力配列の名前が simout とします。こ の場合、simout(1,:) が最初のサンプルに対応し、simout(2,:) が 2 番目のサンプル に対応するといった具合です。入力信号が行列の場合、ワークスペース配列の 3 次元は、指定されたサンプル点での入力信号の値に対応します。たおてば、再 び、simout がワークスペース配列の名前とします。この場合、simout(:,:,1) は最 初のサンプル点の入力信号の値、simout(:,:,2) は 2 番目のサンプル点の入力信号 の値となります。

書き込めるデータの量や書き込めるデータの時間ステップは、ブロックパラメー タで決められます。

- Limit data points to last パラメータは、保存するサンプル点の数を示します。指定した最大値より多くのデータ点がシミュレーションで生成された場合は、 最後に生成されたサンプルが保存されます。すべてのデータを取り込むには、 この値を inf に設定します。
- Decimation パラメータを用いると、n番目のサンプルごとにデータを書き出す ことができます。nは間引きファクタです。デフォルトの間引き値は1で、す べての時間ステップごとにデータを書き出します。
- Sample time パラメータでは、データ点を収集するサンプリング間隔を指定することができます。このパラメータは、時間ステップの間隔が同じでない可変ソルバを使用する場合に有効です。デフォルト値は -1 で、その場合は、ど
の点を書き出すかを決定する際に、接続ブロックからサンプル時間を継承します。

シミュレーション中、ブロックは内部バッファにデータを書き出します。シミュ レーションが完了するかまたは一時停止した場合、そのデータはワークスペース に書き出されます。ブロックアイコンは、データが書き出される行列名を表示し ます。

Structure

この書式は、3 つのフィールド、時間、信号、blockName をもつ構造体です。時間フィールドは空です。blockName フィールドは、To Workspace プロックの名前です。信号フィールドは、2 つのフィールド、値とラベルをもった構造体です。値のフィールドは、信号値の行列です。

Structure with Time

この書式は、シミュレーション時間ステップのベクトルを時間フィールドが含ん でいること以外は、Structure と同じです。

From Workspace ブロックでの保存データの使用

ブロックを使って、書き出されたデータを From Workspace ブロックを用いて別のシミュレーションで " 再生 " しようとする場合、データを保存するフォーマットに Structure with Time を使用してください。

例題

開始時間が 0、Maximum number of sample points i が 100、Decimation が 1、Sample time が 0.5 のシミュレーションで、To Workspace ブロックは、時刻列 0, 0.5, 1.0, 1.5... 秒で最大 100 点を収集します。Decimation を 1 に指定すると、ブロックは ステップ毎にデータを書き出します。

同様の例題で、Maximum number of sample points は 100、Sample time は 0.5 です が、Decimation を 5 とします。この例題で、ブロックは、時間列 0, 2.5, 5.0, 7.5... 秒で最高 100 点まで収集します。Decimation を 5 と指定すると、ブロックは 5 サンプル度にデータを書き出します。サンプル時間によりデータはこれらの点で 確実に書き出されます。

別の例題では、Limit data points to last が3であることを除いてすべてのパラメー タは最初の例題で定義したとおりとします。この場合、最後に収集した3つの行 のみがワークスペースに書き出されます。シミュレーション停止時間が100の場 合、データは99.0秒、99.5秒、および100.0秒(3点)の時間に対応します。 **サポートされて** To Workspace ブロックは、MATLAB ワークスペースに任意の実数または複素数 **いるデータタイ** データタイプの入力をセーブします。 **プ**

パラメータとダ イアログボック ス

Block Parameters: To Workspace 🛛 💌
To Workspace
Write input to specified array or structure in MATLAB's main workspace. Data is not available until the simulation is stopped or paused.
Parameters Variable name:
simout
Limit data points to last:
inf
Decimation:
1
Sample time:
-1
Save format: Structure
OK Cancel Help Apply

Variable name

データを保持する配列名

Limit data points to last

保存する入力サンプルの最大数 (時間ステップあたり1行)。デフォルトは 1000。

Decimation

間引きファクタ。デフォルトは1。

Sample time

データ点を収集するサンプル時間。

Save format

ワークスペースにセーブするシミュレーション出力のフォーマットで、デ フォルトは構造体です。

サンプル時間	継承
次元化	可

目的 線形伝達関数を実現します。

ライブラリ Continuous

詳細

	1	l
1	s+1	ľ

Transfer Fcn ブロックは、入力 (*u*) と出力 (*y*) がつぎの方程式のような伝達関数形式で表現できる伝達関数を実現します。

$$H(s) = \frac{y(s)}{u(s)} = \frac{num(s)}{den(s)} = \frac{num(1)s^{nn-1} + num(2)s^{nn-2} + \dots + num(nn)}{den(1)s^{nd-1} + den(2)s^{nd-2} + \dots + den(nd)}$$

ここで、nn と nd は、それぞれ分子係数と分母係数の数です。num と den は、s の べき乗の降順で表した分子と分母の係数を含んでいます。num はベクトルでも行 列でも構いませんが、den はベクトルでなければなりません。いずれもブロック ダイアログボックス上のパラメータとして指定します。分母の次数は分子の次数 以上でなければなりません。

Transfer Fcn ブロックは、スカラ入力です。ブロックの伝達関数の分子がベクト ルの場合、ブロックの出力もスカラになります。しかし、分子が行列の場合、伝 達関数は入力を分子の行数と同じ幅と等しい出力ベクトルに拡張します。たとえ ば、2行の分子は、スカラ入力とベクトル出力をもつブロックになります。出力 ベクトルの幅は2です。

初期条件は、ゼロにあらかじめ設定されています。初期条件を指定する必要があ る場合、tf2ssを使って状態空間に変換し、State-Space ブロックを使ってくださ い。tf2ss ユーティリティは、システムに対する A、B、C、D 行列を提供します。 詳細については、help tf2ss と入力するか、Control System Toolbox User's Guide を 参照してください。

Transfer Fcn ブロックアイコン

指定した分子と分母に指定した値に依存したアイコンが Transfer Fcn ブロックア イコンに表示されます。

 それぞれを括弧で囲んだ式、ベクトル、変数として指定すると、アイコンは、 指定した係数とsのべき乗をもつ伝達関数を示します。括弧内に変数を指定す ると、その変数は評価されます。たとえば、Numerator を [3,2,1]、 Denominator を ((den))と指定した場合(ここで、(den)は [7,5,3,1])、ブロック アイコンは、つぎのように表示されます。

3s²+2s+1 7s³+5s²+3s+1

それぞれを変数として指定すると、アイコンは変数名とその後に "(s)" を示します。たとえば、Numerator を num、Denominator を den と指定すると、ブロックアイコンは、つぎのように表示されます。



Transfer Fcn ブロックは、double タイプの信号を受け入れ、出力します。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Transfer Fon	×
Matrix expression for numerator, vector expression for denominator. Out width equals the number of rows in the numerator. Coefficients are for descending powers of s.	:put
Parameters	
Numerator:	
[1]	
Denominator: [1 1]	
OK Cancel <u>H</u> elp <u>Apply</u>	

Numerator

分子係数の行ベクトル。複数の出力を生成するために複数の行をもつ行列を 指定することができます。デフォルトは[1]。

Denominator

分母係数の行ベクトル。デフォルトは[11]。

直接フィードスルー	Numerator と Denominator のパラメータの長さが等し い場合のみ
サンプル時間	連続
スカラ拡張	不可
状態	Denominator の長さ -1

次元化 伝達関数の分子が行列の場合、ブロックは、スカラ入 力をベクトル出力に拡張する意味で、可。ブロックの 詳細を参照してください。

ゼロクロッシング なし

目的 設定した時間分だけ入力を遅延させます。

ライブラリ Continuous

詳細

>*2*Q()

Transport Delay ブロックは、指定した時間分だけ入力を遅らせます。これは、時 間遅れをシミュレーションするために使用することができます。

シミュレーションの開始時に、シミュレーション時間が Time delay パラメータ を超えるまで、ブロックは Initial input パラメータを出力し、その後遅れた入力 を出力します。Time delay パラメータは非負でなければなりません。

ブロックは、シミュレーション中に、入力点とシミュレーション時間を、初期サ イズが Initial buffer size パラメータで定義されるバッファに格納します。点数が バッファサイズを超える場合、ブロックは追加メモリを割り当て、シミュレー ションの後で、Simulink は必要なバッファサイズの合計を示すメッセージを表示 します。メモリの割り当てによりシミュレーションが減速するので、シミュレー ション速度が問題になる場合は、このパラメータ値を慎重に定義してください。 遅れ時間が長い場合、このブロックは、特にベクトル化した入力に対して大量の メモリを使用する可能性があります。

格納された入力値の時間に対応しない時間での出力が要求されると、ブロックは データ点の間を線形補間します。遅れがステップサイズより小さい場合、ブロッ クは最後の出力点から外挿するため、不正確な結果を生じることがあります。ブ ロックには直接フィードスルーがないので、現在の入力を用いてその出力値を計 算することはできません。この点を説明するために、ステップサイズが1で現在 時間がt=5の固定ステップシミュレーションを考えます。遅れが0.5の場合、ブ ロックはt=4.5でデータ点を生成する必要があります。最後に格納した時間は t=4 なので、プロックは前進外挿を行います。

Transport Delay ブロックは、離散信号の補間は行わず、*t* - *tdelay* で離散値を出力します。

このブロックは、サンプルヒットのみで出力を遅らせ、保持する Unit Delay ブロックとは異なります。

linmod を使って、Transport Delay ブロックを含むモデルを線形化することには問題があります。この問題を回避する方法の詳細については、第5章の"線形化" を参照してください。

サポートされて Transport Delay ブロックは、double タイプの実数信号を受け入れ、出力します。 いるデータタイ プ パラメータとダ イアログボック ス

Block Parameters: Transport Delay	×
-Transport Delay	
Apply specified delay to the input signal. Best accuracy is achieved when the delay is larger than the simulation step size.	
Parameters	
Time delay:	
Initial input	
0	
Initial buffer size:	
1024	
Pade order (for linearization):	
0	
, , , , , , , , , , , , , , , , , , ,	
OK Cancel Help Apply	

Time delay

出力に伝搬する前に入力信号を遅らせるシミュレーション時間の量。値は非 負でなければなりません。

Initial input

シミュレーションの開始と Time delay との間でブロックが生成する出力。

Initial buffer size

格納するデータ点数に対する初期のメモリの割り当て。

Pade order (for linearization)

線形化ルーチン用のパデ近似次数です。デフォルト値は0です。この結果、 ダイナミック状態のない単ーゲインとなります。次数を、正の整数 n に設定 することは、モデルに n 状態を付け加えますが、線形モデルの精度が上がり ます。

直接フィードスルー	なし
サンプル時間	連続
スカラ拡張	入力と Initial buffer size を除くすべてのパラメータにつ いて
次元化	可
ゼロクロッシング	なし

目的 サブシステムにトリガ端子を追加します。

ライブラリ Signals & Systems

詳細

∡

Trigger ブロックをサブシステムに追加すると、サブシステムは、Triggered サブ システムとなります。Triggered サプシステムは、トリガ端子を通過する信号の 値が指定された方法で(下に説明するように)変化するとき、各積分ステップで 1度だけ実行されます。サブシステムは Trigger ブロックを1つしか含むことが できません。Triggered サブシステムの詳細については、第7章を参照してくだ さい。

Trigger type パラメータでは、サブシステムの実行に対するトリガイベントのタ イプを選択することができます。

- rising を選択すると、トリガ信号が負の方向またはゼロの値から正の値(初期値 が負の場合はゼロ)になる瞬間、サブシステムを起動します。
- fallingを選択すると、トリガ信号が正の方向またはゼロの値から負の値(初期値が正の場合はゼロ)になる瞬間サブシステムを起動します。
- either を選択すると、トリガ信号が正または負いずれかの方向でゼロを横切る 瞬間サブシステムを起動します。
- function-call とすると、S-functionの内部の論理により制御されるサブシステム を起動します(詳細は7章の "Function-Call サブシステム "を参照してください)。

Show output port チェックボックスを選択することによって、トリガ信号を出力 することができます。このオプションを選択することで、システムはトリガの原 因を特定することができます。信号の幅は、トリガ信号の幅です。この信号の値 は、つぎのとおりです。

- トリガ信号が正の方向でゼロを横切る瞬間は1
- トリガ信号が負の方向でゼロを横切る瞬間は-1
- それ以外は0

サポートされて Trigger ブロックは、任意のデータタイプの信号を受け入れます。 いるデータタイ プ

パラメータとダ イアログボック ス

Block Parameters: Trigger	×
Trigger Port	
Place this block in a subsystem to create a triggered subsystem.	
Parameters	
Trigger type:	
Show output port	
Output data type: auto	
OK Cancel Help Apply	

Trigger type

```
サブシステムの実行を決めるトリガイベントのタイプ
```

Show output port

チェックすると、Simulink は Trigger ブロック出力端子を描画し、トリガ信号を出力します。

Output data type

トリガ出力のデータタイプ (double または int8) を指定。auto を選択した場合、Simulink はデータタイプを出力が接続している端子のデータタイプと同じに設定します。端子のデータタイプが double または int8 でない場合は、Simulink はエラーを表示します。

トリガ端子の信号によって決定

特性

サンプル時間 次元化

可

目的 三角関数を適用します。

ライブラリ Math

Trigonometric Function ブロックは、多数の一般的な三角関数を実行します。

> sin >

詳細

Function リストの中から実行する三角関数を選択することができます。Function リストに含まれる三角関数は、sin, cos, tan, asin, acos, atan, atan2, sinh, cosh, tanh で す。Trigonometric Function ブロックは、単一または複数の入力に選択された関数 演算を実行した結果を出力します。

ブロックアイコンには指定された関数の名前が表示されます。Simulink は、適切 な数の入力端子を自動的に描画します。Trigonometric Function ブロックは、 double タイプの実数または複素数の信号を入力として受け入れます。

ベクトル化された出力を生成したい場合には、Fcn ブロックではスカラ出力しか 生成できないので、Trigonometric Function ブロックを使ってください。

Trigonometric Function ブロックは、double タイプの実数または複素数信号を受け

サポートされて いるデータタイ プ

パラメ- イアログ	ータとダ ブボック	r
ス		

Block Parameters: Trigonometric Function			
Trigonometry			
Trigonometric and hyperbolic functions.			
Parameters			
Function: Sin			
Output signal type: auto			
OK Cancel <u>H</u> elp <u>Apply</u>			

Function

三角関数を選択します。

Output signal type

入れ、出力します。

出力される信号のタイプを(複素数または実数)から選択します。

直接フィードスルー	あり
サンプル時間	接続されるブロックから継承
スカラ拡張	関数が2入力を必要とするとき、入力について

次元化 可 ゼロクロッシング なし 目的 一様分布する乱数を生成します。

ライブラリ Sources

詳細



Uniform Random Number ブロックは、設定されたシードを使って、指定した間隔 で一様分布する乱数を生成します。乱数のシードは、シミュレーションが開始さ れる度にリセットされます。生成した乱数列は、繰り返し可能で、同一のシード とパラメータを使って、任意の Uniform Random Number ブロックによって再生で きます。正規分布の乱数を生成するには、Random Number ブロックを使用して ください。

ソルバは、比較的に滑らかな信号を積分するものなので、乱数を積分することは 避けてください。代わりに Band-Limited White Noise ブロックを使用してください。

ブロックの数値パラメータは、スカラ拡張後同じ次元になるようにしてください。Interpret vector parameters as 1-D オプションがオフの場合、ブロックは、パラメータと同じ次元の信号を出力します。Interpret vector parameters as 1-D オプションがオンで数値パラメータが行あるいは列ベクトル(つまり単一行あるいは列の2次元配列)の場合、ブロックは、1ベクトル(1次元配列)信号を出力します。そのほかの場合、ブロックは、パラメータと同じ次元の信号を出力します。

サポートされて いるデータタイ プ Uniform Random Number ブロックは、double タイプの実数信号を出力します。

パラメータとダ イアログボック ス

Block Parameters: Uniform Random Number	ĸ
Uniform Random Number	
Output a uniformly distributed random signal. Output is repeatable for a given seed.	
Parameters	1
Minimum:	
5	
Maximum:	
1	
Initial seed:	
0	
Sample time:	
0	
Interpret vector parameters as 1-D	
OK Cancel Help Apply	

Minimum

間隔の最小値を指定します。デフォルトは-1。

Maximum

間隔の最大値を指定します。デフォルトは1。

Initial seed

乱数発生器に対するシードを設定します。デフォルトは0。

Sample time

特性

サンプル期間。デフォルトは0。

Interpret vector parameters as 1-D

選択すると、Step ブロックの数値パラメータの列あるいは行の秒列の値がベク トル出力信号になります。そうでなければ、ブロックは、パラメータと同じ次元 の信号を出力します。このオプションを選択しない場合、ブロックは、常に、ブ ロックの数値パラメータと同じ次元の信号を出力します。

サンプル時間	連続、	離散または継承
スカラ拡張	不可	
次元化	可	
ゼロクロッシング	なし	

Unit Delay

目的 信号を1サンプル周期だけ遅らせます。

ライブラリ Discrete

詳細
 Unit Delay ブロックは、1 サンプリング間隔だけその入力を遅らせて保持します。
 ブロックに対する入力がベクトルの場合、ベクトルのすべての要素は同じサンプ
 1
 ル遅れ分だけ遅らせます。このブロックは、z-1 離散時間演算子と等価です。

遅れのないサンプルアンドホールド関数が必要な場合は、Zero-Order Hold ブロッ クを使用し、1 単位以上の遅れが必要な場合は、Discrete Transfer Fcn ブロックを 使用してください (Unit Delay ブロックの使用例については、Transport Delay ブ ロックの説明を参照してください)。

サポートされて いるデータタイ リロックは、ユーザ定義のタイプを含むデータタイプの実数または複 素数の信号を入力として受け入れます。入力信号のデータタイプがユーザ定義の ものの場合、初期条件は0でなければなりません。

パラメ-	-タとダ
イアログ	ブボック
ス	

Block Parameters: Unit Delay	×
Unit Delay	
Sample and hold with one sample period delay.	
Parameters	
Initial condition:	
0	
Sample time:	
1	
OK Cancel <u>H</u> elp <u>Apply</u>	

Initial condition

Unit Delay ブロックの出力が不定である間の最初のシミュレーション周期に 対するブロック出力。このパラメータを慎重に選択することによって、この 周期における望ましくない出力挙動を最小限に抑えることができます。デ フォルトは 0。

Sample time

サンプル間の時間間隔。デフォルトは1。

直接フィードスルー	なし
サンプル時間	離散
スカラ拡張	Initial condition パラメータまたは入力について

状態 接続されるブロックまたはパラメータから継承

可

次元化

ゼロクロッシング なし

目的 可変時間分だけ入力を遅らせます。

ライブラリ Continuous

詳細



Variable Transport Delay ブロックは、可変時間遅れをシミュレーションするため に使用することができます。このブロックは、パイプ内の液体をポンプで操作す るモータ速度が可変であるような、パイプをもつシステムをモデル化するために 使用することができます。

ブロックは2つの入力を受け入れます。最初の入力はブロックを通過する信号、 2番目の入力は、つぎのアイコンに示すような時間遅れです。



Maximum delay パラメータは、時間遅れ入力をもつことのできる最大値を定義 します。ブロックはこの値を超える遅れの値をクリップします。Maximum delay は、ゼロ以上でなければなりません。時間遅れが負になると、ブロックはそれを ゼロとして扱い、ワーニングメッセージを表示します。

シミュレーション中、ブロックは時間と入力値を組合わせて内部バッファに格納 します。シミュレーションを開始してからシミュレーション時間が時間遅れ入力 を超えるまで、Initial input パラメータを出力します。その後、各シミュレー ションステップ毎に、ブロックは現在のシミュレーション時間から遅れ時間を差 し引いた時間に対応する時間で信号を出力します。

格納された入力値の時間に対応しない時間での出力が必要な場合、ブロックは データ点の間を線形補間します。時間遅れがステップサイズより小さい場合、ブ ロックは出力点を外挿します。この結果、精度が低くなる可能性があります。ブ ロックは、この端子に直接フィードスルーをもたないので、現在の入力を用いて その出力値を計算することはできません。この点を示すために、ステップサイズ が1で現在時間がt=5である、固定ステップシミュレーションを考えます。遅 れが0.5の場合、ブロックはt=4.5でデータ点を生成する必要があります。最後 に格納した時間はt=4なので、ブロックは前進外挿を実行します。

Variable Transport Delay ブロックは、離散信号の補間は行いません。その代わり に、*t*-*tdelay* で離散値を出力します。 サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

特性

Variable Transport Delay ブロックは、double タイプの実数信号を受け入れ、出力します。

Block Parameters: Variable Transport Delay 💌
Apply a delay to the first input signal. The second input specifies the delay time. Best accuracy is achieved when the delay is larger than the simulation step size.
Parameters Maximum delay:
Initial input:
0
Buffer size:
1024
Pade order (for linearization):
0
OK Cancel Help Apply

Maximum delay

時間遅れ入力の最大値。値を負にすることはできません。デフォルトは10。

Initial input

シミュレーション時間が最初に時間遅れ入力を超えるまでブロックが生成する出力。デフォルトは 0。

Buffer size

ブロックが格納できる点数。デフォルトは 1024。

Pade order (for linearization)

線形化ルーチンのためのパデ近似次数。デフォルト値は、0です。この結果、ダイナミック状態のない均一ゲインとなります。次数に正の整数 n を設定するとモデルに n 状態が加えられますが、transport delayの線形モデルの精度が上がります。

直接フィードスルー あり、時間遅れ (2番目の) 入力について

サンプル時間 連続

スカラ拡張 入力と Buffer size を除くすべてのパラメータについて

次元化 可 ゼロクロッシング なし 目的 入力ベクトルの幅を出力します。

ライブラリ Signals & Systems

Width ブロックは、その入力ベクトルの幅を出力として生成します。



詳細

プ

サポートされて Width ブロックは、任意のデータタイプの実数または複素数信号値を受け入れま す。この場合、同じタイプでなく、種々のタイプが組合わさっていても構いませ いるデータタイ ん。Width ブロックは、double タイプの実数信号を受け入れ、出力します。

パラメータとダ イアログボック ス	Block Parameters: Width Width Output the width of the input sign OK Cancel	al.	
特性	サンプル時間	一定	
	次元化	可	

XY Graph

目的 MATLAB の Figure ウィンドウを使って、信号の X-Y プロットを表示します。

ライブラリ Sinks

詳細

) O XY Graph ブロックは、MATLAB の Figure ウィンドウに、その入力の X-Y プロットを表示します。

ブロックには2つのスカラ入力があります。ブロックは最初の入力(x方向) データに対して2番目の入力(y方向)データをプロットします。このブロック は、リミットサイクルと他の2状態データを調べるのに役立ちます。指定範囲外 のデータは表示されません。

Simulink は、シミュレーションの開始時に、モデル内の各 XY Graph ブロックに 対する Figure ウィンドウを開きます。

XY Graph ブロックの使い方を示すデモを見るために、コマンドウィンドウに lorenzs と入力してください。

XY Graph ブロックは、double タイプの実数信号を受け入れます。

サポートされて いるデータタイ プ

パラメータとダ イアログボック ス

biock Falanieleis. Al Graph
XY scope. (mask) (link)
$X\!Y$ scope using MATLAB graph window. First input is used as time base. Enter plotting ranges.
Parameters
x-min:
-1
x-max:
1
y-min:
·1
y-max:
1
Sample time:
-1
OK Cancel <u>H</u> elp <u>Apply</u>

x-min

x軸の最小値。デフォルトは-1。

x-max

x軸の最大値。デフォルトは1。

y-min

y軸の最小値。デフォルトは-1。

y-max

y軸の最大値。デフォルトは1。

0

Sample time

サンプル間の時間間隔。デフォルトは-1 で、サンプル時間が接続ブロック で決まることを意味します。

接続されるブロックから継承

特性

サンプル時間 状態 目的 1 サンプル周期のゼロ次ホールドを実現します。

ライブラリ Discrete

詳細



Zero-Order Hold ブロックは、指定したサンプリングレートで動作するサンプルア ンドホールド関数を実現します。ブロックは1つの入力を受け入れ、1つの出力 を生成しますが、いずれもスカラまたはベクトルとすることができます。

Zero-Order Hold ブロックは1つまたは複数の信号を離散化したり、信号を異なる レートでリサンプリングしたりするための仕組みを提供します。サンプリングを モデル化する必要があるような場合に、他のより複雑な離散関数ブロックを必要 とすることなく、このブロックを使用することができます。たとえば、Quantizer ブロックと組み合わせて、入力アンプをもつ A/D 変換器をモデル化するために 使用することができます。

サポートされて A Zero-Order Hold ブロックは、任意のデータタイプの実数または複素数信号を受いるデータタイ け入れます。 プ

パラメータとダ イアログボック ス

Block Parameters: Zero-Urder Hold	×
Zero-Order Hold	
Zero-order hold.	
Parameters	5
Sample time:	
1	
OK Cancel Help Apply	

Sample time

サンプル間の時間間隔。デフォルトは1。

直接フィードスルー	あり
サンプル時間	離散
スカラ拡張	可
States	0
次元化	可
ゼロクロッシング	なし

目的 極と零点について指定した伝達関数を実現します。

ライブラリ Continuous

詳細

> (s-1) s(s+1) Zero-Pole ブロックは、Laplace 演算子 s を使って、指定した零点、極、ゲインを もつシステムを実現します。

伝達関数は、因数分解型または零点 - 極 - ゲイン型で表すことができますが、 MATLAB では、単入力単出力システムは、つぎのようになります。

$$H(s) = K \frac{Z(s)}{P(x)} = K \frac{(s - Z(1))(s - Z(2))\dots(s - Z(m))}{(s - P(1))(s - P(2))\dots(s - P(n))}$$

ここで、Z は零点ベクトル、P は極ベクトル、K はゲインを表します。Z はベク トルでも行列でも構いませんが、P はベクトルでなければなりません。K は長さ が Z の行数に等しいスカラまたはベクトルです。極の数は零点の数以上でなけれ ばなりません。極と零点が複素数の場合、それらは複素共役対でなければなりま せん。

ブロック入力と出力幅は、零点行列の行数と同じです。

Zero-Pole ブロックアイコン

Zero-Pole ブロックは、パラメータをどのように指定するかに応じて、伝達関数 をアイコンに表示します。

 それぞれを式かベクトルとして指定すると、アイコンは指定した零点、極、 ゲインをもつ伝達関数を示します。括弧内に変数を指定すると、その変数を 求めて表示します。

たとえば、**Zeros** を [3,2,1]、**Poles** を (poles)(ただし、poles はワークスペース内 で [7,5,3,1] と定義されています)、**Gain** を gain と指定すると、アイコンはつぎ のように表示されます。

ĺ	gain(s-3)(s-2)(s-1)
1	(\$7)(\$5)(\$3)(\$1)

それぞれを変数として指定すると、アイコンは、変数名と該当する場合はその後に "(s)" を表示します。たとえば、Zeros を zeros、Poles を poles、Gain を gain と指定すると、アイコンはつぎのように表示されます。



サポートされて いるデータタイ プ

パラメータとダ	
イアログボック	
ス	

Block Parameters: Zero-Pole
Zero-Pole
Matrix expression for zeros. Vector expression for poles and gain. Output width equals the number of columns in zeros matrix, or one if zeros is a vector.
Parameters
Zeros:
[1]
Poles:
[0 -1]
Gain:
[1]
OK Cancel <u>H</u> elp <u>Apply</u>

Zero-Pole ブロックは、double タイプの実数信号を受け入れます。

Zeros

零点の行列。デフォルトは[1]。

Poles

極のベクトル。デフォルトは[0-1]。

Gain

利得のベクトル。デフォルトは[1]。

直接フィードスルー	Poles と Zeros のパラメータの長さが同じ場合のみ
サンプル時間	連続
スカラ拡張	不可
状態	Poles ベクトルの長さ
次元化	不可
ゼロクロッシング	なし

10

モデル構築コマンド

はじめに .																				.10-2
コマンドにハ	٤E	ラン	κ-	- /	77	Έł	訂	ÈЗ	する	5 7	53	£								10-3
Simulink オブ	シ	Ľз	こち	7	- 12	こり	パフ	くを	E捎	訂	Ξġ	t a	らた	5Ż	F					10-3
add_block .																				10-4
add_line																				10-5
bdclose																				10-6
bdroot																				10-7
close_system																				10-8
delete_block																				10-10
delete_line .																				10-11
find_system																				10-12
gcb																				10-17
gcbh																				10-18
gcs																				10-19
get_param .																				10-20
new_system																				10-22
open_system																				10-23
replace_block																				10-24
save_system																				10-26
set_param .																				10-27
simulink																				10-29

はじめに

つぎの表には、本章で説明するコマンドで実行される作業を示しています。本章 のリファレンス節には、コマンドをアルファベット順にリストしています。

作業内容	コマンド
新しい Simulink システムを作成します	new_system
既存のシステムを開きます	open_system
システムウィンドウを閉じます	close_system, bdclose
システムを保存します	save_system
システム、ブロック、ライン、注釈を検索し ます	find_system
新しいブロックをシステムに追加します	add_block
システムからブロックを削除します	delete_block
システム内のブロックを置き換えます	replace_block
システムにラインを追加します	add_line
システムからラインを削除します	delete_line
パラメータ値を取得します	get_param
パラメータ値を設定します	set_param
カレントブロックのパス名を取得します	gcb
カレントシステムのパス名を取得します	gcs
カレントブロックのハンドルを取得します	gcbh
最上位の Simulink システムの名前を出力しま す	bdroot
Simulink ブロックライブラリを開きます	simulink

コマンドにパラメータを指定する方法

本章で説明するコマンドでは、システム、ブロック、またはブロックパラメータ を記述する引数を指定する必要があります。付録 A の " モデルとブロックパラ メータ " には、モデルとブロックパラメータの包括的な表が示してあります。

Simulink オブジェクトにパスを指定する方法

本章で説明するコマンドの多くは、Simulinkのシステムまたはブロックを識別す る必要があります。それらのパスを指定することによって、システムとブロック を識別してください。

- システムを識別するためには、その名前を指定します。これは、mdl 拡張子を 除いたシステム記述を含むファイル名です。 system
- サブシステムを識別するためには、システムとサブシステムが存在するサブシステムの階層を指定します。

system/subsystem1/.../subsystem

ブロックを識別するためには、ブロックを含むシステムのパスを指定し、ブロック名を指定します。

system/subsystem₁/.../subsystem/block

ブロック名にニューラインまたはキャリッジリターンが含まれる場合、ブロック 名を文字列ベクトルとして指定し、sprintf('\n') をニューラインキャラクタとして 使ってください。たとえば、以下のコードは、ニューラインキャラクタを cr に 割り当て、Signal Generator ブロックの Amplitude パラメータに対する値を取り 出します。

```
cr = sprintf(\n');
get_param(['untitled/Signal',cr,'Generator'],'Amplitude')
ans =
1
```

ブロック名にスラッシュキャラクタ (/) が含まれる場合は、ブロック名を指定す るときにスラッシュを繰り返します。たとえば、mymodel システム内の Signal/ Noise という名前のブロックに対する Location パラメータの値を取り出すには、 つぎのように入力します。

get_param('mymodel/Signal//Noise','Location')

目的	ブロックを Simulink システムに追加します。
表示	add_block('src', 'dest') add_block('src', 'dest', 'parameter1', value1,)
詳細	add_block('src', 'dest') は、絶対パス名 'src' のブロックを絶対パス名 'dest' の新しい ブロックにコピーします。新しいブロックのブロックパラメータは、オリジナル のブロックパラメータと一致します。名前 'built-in' は、すべての Simulink 組み込 みブロック (マスクされたブロックではない Simulink ブロックライブラリで利用 可能なブロック) に対してソースシステム名として使用することがきます。
	add_block('src', 'dest_obj', 'parameter1', value1,) は、名前を指定されたパラメータ が指定の値をもつ、上記のようなコピーを作成します。追加引数はすべて、パラ メータと値の組合わせで指定しなければなりません。
例題	つぎのコマンドは、Scope ブロックを simulink システムの Sinks サブシステムか ら、engine システムの timing サブシステム内の Scope1 という名前のブロックに コピーします。
	add_block('simulink/Sinks/Scope', 'engine/timing/Scope1')
	つぎのコマンドは、F14 システム内に controller という名前の新しいサブシステ ムを作成します。
	add_block('built-in/SubSystem', 'F14/controller')
	つぎのコマンドは、組み込み Gain ブロックを mymodel システムの Volume という名前のブロックにコピーし、Gain パラメータに値 4 を割り当てます。 add_block('built-in/Gain', 'mymodel/Volume', 'Gain', '4')

delete_block, set_param

参考

目的 ラインを Simulink システムに追加します。

表示 $h = add_line('sys', 'oport', 'iport')$

 $h = add_line('sys', points)$

詳細 add_line コマンドは、指定したシステムにラインを追加し、ハンドルを新しいラ インに返します。ラインは、つぎの2つの方法で定義することができます。

- ラインによって接続されるブロックの端子の名前を指定
- ラインセグメントを定義する点の位置を指定

add_line('sys', 'oport', 'iport') は、システムに対する直線を、指定したブロック出力 端子 'oport' から指定したブロック入力端子 'iport' に追加します。'oport' と 'iport' は、ブロック名と端子識別子から構成される 'block/port' 形式の文字列です。ほと んどのブロック端子は、上から下または左から右へ端子を番号付けすることに よって、たとえば、'Gain/1' や 'Sum/2' のように指定します。Enable、Trigger、 State 端子は、'subsystem_name/Enable'、'subsystem_name/Trigger' または 'Integrator/ State' のように名前で識別されます。

add_line(system, points) は、セグメント化したラインをシステムに追加します。 points 配列の各行は、ラインセグメント上の点の x 座標と y 座標を指定します。 原点はウィンドウの左上隅にあります。信号は最初の行で定義した点から最終行 で定義した点まで流れます。新しいラインの開始が既存のブロックの出力または ラインに近い場合に接続が行われます。同様に、ラインの終端が既存の入力に近 い場合に接続が行われます。

つぎのコマンドは、Sine Wave ブロックの出力を Mux ブロックの最初の入力に接 続するように、mymodel システムにラインを追加します。

add_line('mymodel','Sine Wave/1','Mux/1')

つぎのコマンドは、(20,55)から(40,10)、(60,60)まで伸びるラインを mymodel シ ステムに追加します。

add_line('mymodel',[20 55; 40 10; 60 60])

参考 delete_line

例題

bdclose

目的	任意の、あるいはすべての Simulink システムウィンドウを無条件に閉じます。
表示	bdclose bdclose('sys') bdclose('all')
詳細	引数なしの bdclose は、現在のシステムウィンドウを無条件かつ確認なしに閉じ ます。最後に保存した後で行われたシステムの変更はすべて失われます。 bdclose('sys') は、指定のシステムウィンドウを閉じます。 bdclose('all') は、すべてのシステムウィンドウを閉じます。
例題	つぎのコマンドは、vdp システムを閉じます。 bdclose('vdp')
参考	close_system, new_system, open_system, save_system

目的	最上位の Simulink システムの名前を出力します。
表示	bdroot bdroot('obj')
詳細	引数なしの bdroot は、最上位のシステムの名前を出力します。 bdroot('obj') は、'obj' がシステムまたはブロックのパス名であるとき、指定したオ ブジェクト名を含む最上位システムの名前を出力します。
例題	つぎのコマンドは、カレントのブロックを含む最上位システムの名前を出力しま す。 bdroot(gcb)
参考	find_system, gcb

目的 Simulink システムウィンドウまたはブロックダイアログボックスを閉じます。 表示 close_system close_system('sys') close_system('sys', saveflag) close_system('sys', 'newname') close_system('blk') 詳細 引数なしの close_system は、カレントのシステムまたはサブシステムウィンドウ を閉じます。カレントのシステムが最上位システムであり、それに対する修正が 行われている場合、close systemは、メモリから除去する前に、変更したシステ ムをファイルに保存するかどうかを尋ねます。カレントのシステムについては、 gcs コマンドの説明で定義してあります。 close system('sys') は、指定したシステムまたはサブシステムウィンドウを閉じま す。 close_system('sys', saveflag) は、指定した最上位システムウィンドウを閉じて、それ をメモリから除去します。 • saveflag が0の場合、システムは保存されません。 saveflag が1の場合、システムはカレントの名前で保存されます。 close_system('sys', 'newname')は、指定した最上位システムを指定した新しい名前で ファイルに保存してから、システムを閉じます。 close_system('blk')は、'blk' がブロックの絶対パス名であるとき、指定したブロック に関連するダイアログボックスを閉じるか、ブロックの CloseFcn コールバック パラメータが定義されていればそれを呼び出します。その他の引数はすべて無視 されます。 例題 つぎのコマンドは、カレントのシステムを閉じます。 close_system つぎのコマンドは、vdp システムを閉じます。 close_system('vdp') つぎのコマンドは、engine システムをそのカレントの名前で保存してから、それ を閉じます。 close_system('engine', 1)

つぎのコマンドは、mymdl12システムを testsys という名前で保存してから、それを閉じます。

close_system('mymdl12', 'testsys')

つぎのコマンドは、engine システムの Combustion サブシステムにおける Unit Delay ブロックのダイアログボックスを閉じます。

close_system('engine/Combustion/Unit Delay')

bdclose, gcs, new_system, open_system, save_system

参考

目的 Simulink システムからブロックを削除します。

表示 delete_block('blk')

- **詳細** delete_block('blk') は、'blk' がブロックの絶対パス名であるとき、システムから指定 したブロックを削除します。
- 例題 つぎのコマンドは、vdp システムから Out1 ブロック を除去します。 delete_block('vdp/Out1')

参考 add_block

- 目的 システムからラインを削除します。
- 表示 delete_line('sys', 'oport', 'iport')
- 詳細 delete_line('sys', 'oport', 'iport') は、指定したブロック出力端子 'oport' から指定した ブロック入力端子 'iport' までのラインを削除します。'oport' と 'iport' は、ブロック 名と端子識別子から構成される 'block/port' 形式の文字列です。ほとんどのブロッ ク端子は、'Gain/1' や 'Sum/2' のように、上から下または左から右に番号付けする ことによって識別します。Enable、Trigger、および State 端子は、'subsystem_name/Enable'、'subsystem_name/Trigger'、'Integrator/State' のように名前 で識別されます。

delete_line('sys', [x y]) は、指定した点 (x,y) を含むシステムのラインの 1 つが存在 すれば、それを削除します。

例題 つぎのコマンドは、Sum ブロックを Mux ブロックの2番目の入力に接続しているラインを mymodel システムから削除します。

delete_line('mymodel','Sum/1','Mux/2')

参考 add_line

詳細

目的 システム、ブロック、ライン、注釈を検索します。

表示 find_system(sys, 'c1', cv1, 'c2', cv2,...'p1', v1, 'p2', v2,...)

find_system(sys, 'c1', cv1, 'c2', cv2,...'p1', v1, 'p2', v2,...) は、sys で指定された単数また は複数のシステムやサブシステムを c1, c2 等によって指定された制約条件を用い て検索し、指定されたパラメータ値 v1, v2 等をもつオブジェクトに対するハンド ルや絶対パスを出力します。sys にはパス名(または複数のパス名からなるセル 配列)、ハンドル(または複数のハンドルからなるベクトル)の双方を指定でき ますし、入力を省略することもできます。sys がパス名またはパス名からなるセ ル配列に場合は、find_system は検索によって発見したオブジェクトに対する複数 のパス名からなるセル配列を出力します。sys はハンドルまたはハンドルからな るベクトルの場合は、find_system は発見したオブジェクトに対する複数のハン ドルからなるベクトルを出力します。sys が省略された場合は、find_system はす べてのオープンシステムを検索しパス名からなるセル配列を出力します。

パラメータ名については、大文字と小文字の区別はありません。値の文字列については、デフォルトで大文字と小文字を区別します(詳細は、'CaseSensitive'サーチの制約を参照)。ダイアログボックスのエントリに対応するパラメータは、文字列の値をもちます。モデルとブロックパラメータのリストについては、付録Aの"モデルとブロックパラメータ"を参照してください。

名前	値のタイプ	詳細
'SearchDepth'	スカラ	指定されたレベルに検索の深さを 限定します(0オープンシステム のみ検索します。1最上位システ ムを構成するブロック、サブシス テムのみ検索します。2最上位シ ステムとその子システムを検索し まする)。デフォルトは、全てのレ ベルを設定します。
'LookUnderMasks'	'none'	マスクされたブロックを検索しま せん。

検索の制約条件として、つぎのいずれかを指定することができます。
名前	値のタイプ	詳細
	{'graphical'}	ワークスペースやダイアログを含 まないマスクされたブロックも検 索します。デフォルトです。
	'functional'	ダイアログをもたないマスクされ たブロックも検索します。
	'all'	すべてのマスクされたブロックも 検索します。
'FollowLinks'	'on' {'off'}	'on' の場合は、検索はライブラリブ ロックヘリンクされたものを含み ます。デフォルトは、'off' です。
'FindAll'	'on' {'off'}	'on' の場合、システムのラインや端 子、注釈に検索の対象が拡張され ます。デフォルトは、'off' です。 find_system は、このオプションが 'on' のときは sys の配列タイプにか かわらず、ハンドルからなるベク トルを出力します。
'CaseSensitive'	{'on'} 'off'	'on'の場合、文字列もマッチングに おいて検索は大文字と小文字の区 別を行います。デフォルトは 'on' で す。
'RegExp'	'on' {'off'}	'on'の場合、検索表現を正規表現と して取り扱います。デフォルトは 'off'です。

表では、括弧内にデフォルトの制約の値を示しています。'constraint' が省略されると、find_system はデフォルトの制約値を利用します。

例題

つぎのコマンドは、すべてのオープンシステムとブロックの名前を含むセル配列 を出力します。

find_system

つぎのコマンドは、開いているすべてのブロック線図の名前を出力します。

open_bd = find_system('type', 'block_diagram')

つぎのコマンドは、clutch システム内の Unlocked サブシステムの子であるすべての Goto ブロックの名前を出力します。

find_system('clutch/Unlocked','SearchDepth',1,'BlockType','Goto')

つぎのコマンドは、Gain パラメータの値が1 であるような vdp システム内のす べての Gain ブロックの名前を出力します。

gb = find_system('vdp', 'BlockType', 'Gain') find_system(gb, 'Gain', '1')

上記のコマンドは、つぎのコマンドと等価です。

find_system('vdp', 'BlockType', 'Gain', 'Gain', '1')

これらのコマンドは、vdp システム内のすべてのラインと注釈のハンドルを出力 します。

sys = get_param('vdp', 'Handle');

l = find_system(sys, 'FindAll', 'on', 'type', 'line');

a = find_system(sys, 'FindAll', 'on', 'type', 'annotation');

正規表現の検索

RegExp'制約を 'on'と指定した場合、find_system はサーチ文字列を正規表現とし て取り扱います。正規表現とは、キャラクタが特殊なパターンマッチングの意味 をもつ文字列です。たとえば、ピリオド(.)は正規表現ではそれ自身だけでなく それ以外のキャラクタとマッチします。

正規表現は、find_systemを使って行うことができる検索のタイプを大きく拡張 します。たとえば、正規表現を使って部分ワードサーチを行うことができます。 指定したキャラクタからなる文字列を含む、または先頭および末尾にもつパラ メータをもつすべてのオプジェクトに対する検索を行うことができます。

正規表現を効果的に利用するためには、正規表現に含まれる特殊キャラクタの意味を知っておく必要があります。つぎの表は、find_subystem がサポートする特殊キャラクタをリストし、それらの使用法を説明します。

表現	使用法
	任意のキャラクタとマッチ。たとえば、文字列 'a.' は 'aa', 'ab', 'ac' 等とマッチします。
*	そのキャラクタの前にある0個およびそれ以上のキャラクタ がマッチ。たとえば、'ab*'は'a','ab','abb'等とマッチします。 表現'.*'は、空文字列を含む任意の文字列とマッチします。
+	そのキャラクタの前にある 1 個およびそれ以上のキャラクタ がマッチ。たとえば、'ab+' は 'ab', 'abb' 等とマッチします。
^	文字列の先頭がマッチ。たとえば、'^a.*' は、'a' で始まる任意 の文字列とマッチします。
\$	文字列の末尾がマッチ。たとえば、'.*a\$'は 'a' で終わる任意 の文字列とマッチします。
	そのつぎのキャラクタを通常キャラクタとして取り扱いま す。"エスケープ"キャラクタは正規表現を特殊キャラクタ を含む表現とマッチさせます。たとえば、文字列 \\'は\キャ ラクタを含む任意の文字列とマッチします。
[]	指定した複数のキャラクタのうちのいずれか1つがマッチ。 たとえば、f[oa]r'は、for'および far'とマッチします。キャ ラクタの中には、括弧ないで特殊な意味をもつものがありま す。ハイフン(-)は、マッチするキャラクタの範囲を示しま す。たとえば、'[a-zA-Z1-9]'は、任意の英数字キャラクタが マッチします。circumflex(^)は、マッチングを行わないキャ ラクタを示します。たとえば、'f[^i]r'は far'や for'とはマッ チしますが、'fir'とはマッチしません。
\w	ワードキャラクタがマッチ (これは、[a-z_A-Z0-9] に対する 省略表現です)。たとえば、 'n∖w'は 'nu'とマッチしますが、 'μ'とはマッチしません。
\d	任意の数字がマッチ ([0-9] の省略表現)。たとえば、 ⁄\d+' は任 意の整数とマッチします。
\D	非数字がマッチ ([^0-9] の省略表現)。

表現	使用法
\s	空白がマッチ ([\t\r\n\f] の省略表現)。
\S	空白でないキャラクタがマッチ ([^ \t/r\n\f] の省略表現)。
\ <word\></word\>	正確に WORD とマッチ。ここで、WORD は他の単語とは空 白で区切られたキャラクタからなる文字列です。たとえば、 \ <to\>`は `to`とマッチしますが、`today`とはマッチしません。</to\>

Simulink システムの検索に正規表現を使うためには、'regexp' 検索制約条件を find_system コマンド内で'on'と指定し、通常の検索文字列で使う位置で正規表現 を使います。

たとえば、つぎのコマンドは Simulink に付属している clutch モデルのすべての inport および outport ブロックを検索します。

find_system('clutch', 'regexp', 'on', 'blocktype', 'port')

get_param, set_param

参考

gcb

目的 カレントの Simulink ブロックの絶対パス名を取り出します。

表示 gcb

gcb('sys')

詳細 gcb は、カレントのシステム内のカレントのブロックの絶対パス名を出力します。

gcb('sys')は、指定したシステム内のカレントのブロックの絶対パス名を出力します。

カレントのブロックとは、つぎのいずれかです。

- ・ 編集中では、最後にクリックしたブロックがカレントのブロックです。
- S-Function ブロックを含むシステムのシミュレーション中では、現在その対応 する MATLAB 関数を実行している S-Function ブロックがカレントのブロック です。
- コールバック中では、そのコールバックルーチンが実行されているブロック がカレントのブロックです。
- MaskInitialization 文字列の評価中では、そのマスキングが評価されているブロックがカレントのブロックです。

つぎのコマンドは、最後に選択したブロックのパスを出力します。

gcb ans = clutch/Locked/Inertia

つぎのコマンドは、カレントのブロックの Gain パラメータの値を取り出します。

```
get_param(gcb,'Gain')
ans =
1/(Iv+Ie)
```

参考 gcbh, gcs

例題

目的 カレントブロックのハンドルを取得します。

表示 gcbh

詳細 gcbh は、カレントシステムのカレントブロックのハンドルを取得します。

ユーザは、このコマンドを、親システムをもたないブロックの識別や割り当てに 使うことができます。このコマンドは、ブロックセットの開発者にとって有用な ものです。

例題 つぎのコマンドは、最も新しく選択したブロックのハンドルを返します。

gcbh

gcb

ans =

281.0001

参考

目的 カレントシステムのパス名を取得します。

表示 gcs

詳細 gcsは、カレントのシステムの絶対パス名を出力します。

カレントのシステムとは、つぎのいずれかです。

- 編集中では、最後にクリックしたシステムまたはサブシステムがカレントのシステムです。
- S-Function ブロックを含むシステムのシミュレーションの実行中では、現在評価されている S-Function ブロックを含むシステムまたはサブシステムがカレントのシステムです。
- コールバック中では、そのコールバックルーチンが実行されている任意のブロックを含むシステムがカレントのシステムです。
- MaskInitialization 文字列の評価中では、そのマスキングが評価されているブロックを含むシステムがカレントのシステムです。

カレントのシステムは、常にカレントのモデルまたはカレントのモデルのサブシ ステムです。カレントのモデルを取得するには、bdrootを使ってください。

例題

つぎの例は、最後に選択したブロックを含むシステムのパスを出力します。

gcs ans =

clutch/Locked

参考 gcb, bdroot

目的	システムとブロックのパラメータ値を取得します。
----	-------------------------

- 表示 get_param('obj', 'parameter') get_param({ objects }, 'parameter') get_param(handle, 'parameter') get_param('obj', 'ObjectParameters') get_param('obj', 'DialogParameters')
- **詳細** get_param('obj', 'parameter') は、'obj' がシステムまたはブロックのパス名であると き、指定したパラメータの値を出力します。パラメータ名については大文字と文 字の区別はありません。

get_param({objects}, 'parameter')は、絶対パスで指定されたセル配列を入力として受け入れ、そのセル配列内で指定されたオブジェクト全てに共通なパラメータ値の取得を可能にします。

get_param(handle, 'parameter') は、handle で指定されたハンドルをもつオブジェクトの指定されたパラメータを出力します。

get_param('obj', 'ObjectParameters') は、obj パラメータ値を示す構造体を出力しま す。出力される構造体の各フィールドは、特定のパラメータに対応し、パラメー タ名をもっています。たとえば、Name フィールドは、オブジェクトの Name パ ラメータに対応します。各パラメータフィールド自体は、Name,Type,Attributes の3つのフィールドを含んでおり、これら3つのフィールドは、各々、パラメー タの名前(例: "Gain")、データタイプ(例:文字列)、属性(例:読み込み専 用)を指定します。

get_param('obj', 'DialogParameters') は、指定されたブロックのダイアログパラメー タ名を含むセル配列を出力します。

付録 A の "モデルとブロックパラメータ "に、モデルとブロックパラメータのリ ストがあります。

つぎのコマンドは、clutch システムの Requisite Friction サブシステムにおける Inertia ブロックに対する Gain パラメータの値を出力します。

get_param('clutch/Requisite Friction/Inertia','Gain')
ans =
1/(Iv+Ie)

例題

つぎのコマンドは、7-3ページの"マスクされたサンプルサブシステム"に記述された mx+b システム(この場合、カレントシステム)の中のすべてのブロックの ブロックタイプを表示します。

blks = find_system(gcs, Type', 'block'); listblks = get_param(blks, 'BlockType')

listblks =

'SubSystem' 'Inport' 'Constant' 'Gain' 'Sum' 'Outport'

つぎのコマンドは、カレントに選択したブロックの名前を出力します。

get_param(gcb, 'Name')

つぎのコマンドは、カレントに選択したブロックの Name パラメータに関連した ものを取得します。

p = get_param(gcb, 'ObjectParameters');

a = p.Name.Attributes

ans = 'read-write' 'always-save'

つぎのコマンドは、Sine Wave ブロックのダイアログパラメータを取得します。

p = get_param('untitled/Sine Wave', 'DialogParameters')

- p =
 - 'Amplitude' 'Frequency' 'Phase' 'SampleTime'

find_system, set_param

参考

10-21

- 目的 新しい Simulink システムを作成します。
- 表示 new_system('sys')
- 詳細 new_system('sys')は、指定した名前で新しい空のシステムを作成します。'sys'が パスを指定している場合、新しいシステムはパスで指定したシステムのサブシス テムとなります。new_systemはシステムウィンドウを開きません。

新しいシステムに対するデフォルトのパラメータ値のリストについては、付録 A の"モデルとブロックパラメータ"を参照してください。

例題 つぎのコマンドは、'mysys' という名前の新しいシステムを作成します。

new_system('mysys')

つぎのコマンドは、vdp システムに 'mysys' という名前の新しいサブシステムを作成します。

new_system('vdp/mysys')

参考 close_system, open_system, save_system

目的	Simulink システムウィンドウまたはブロックダイアログボックスを開きます。
表示	open_system('sys') open_system('blk') open_system('blk', 'force')
詳細	open_system('sys') は、指定したシステムまたはサブシステムウィンドウを開きま す。
	open_system('blk') は、'blk' がブロックの絶対パス名であるとき、指定したブロック に関連するダイアログボックスを開きます。ブロックの OpenFcn コールバック パラメータが定義されていれば、ルーチンが評価されます。
	open_system('blk', 'force') は、'blk' がブロックの絶対パス名かマスクされたシステム であるとき、指定したシステムのマスクの下を表示します。このコマンドは、 Look Under Mask メニューアイテムを使用するのと同じ結果になります。
例題	つぎのコマンドは、デフォルトのスクリーン位置に controller システムを開きます。
	open_system('controller')
	つぎのコマンドは、controller システム内の Gain ブロックに対するブロックダイ アログボックスを開きます。
	open_system('controller/Gain')
参考	close_system, new_system, save_system

目的 Simulink モデル内のブロックを置き換えます。

表示 replace_block('sys', 'blk1', 'blk2', 'noprompt') replace_block('sys', 'Parameter', 'value', 'blk', ...)

詳細 replace_block('sys', 'blk1', 'blk2') は、ブロックまたはマスクタイプが 'blk1' である 'sys' 内のすべてのブロックを 'blk2' で置き換えます。'blk2' が Simulink の組み込み ブロックの場合は、ブロック名のみで構いません。'blk' が別のシステムにもある 場合は、そのブロックの絶対パス名が必要です。'noprompt' を省略すると、 Simulink は、置換する前に、一致するブロックを選択するよう尋ねるダイアログ ボックスを表示します。'noprompt' 引数を指定すると、ダイアログボックスは表 示されなくなります。出力変数を指定すると、置き換えられたブロックのパスが その変数に格納されます。

replace_block('sys', 'Parameter', 'value', ..., 'blk') は、指定したパラメータに対して指定した値をもつ 'sys' 内のすべてのブロックを 'blk' で置き換えます。パラメータと値の組合わせを何組でも指定することができます。

注意 このコマンドが行う変更を元に戻すことは困難な可能性があるので、まず システムを保存することをお薦めします。

例題

つぎのコマンドは、f14 システムにあるすべての Gain ブロックを Integrator ブ ロックで置き換え、置き換えられたプロックのパスを RepNames に格納します。 Simulink は、置換を行う前に、一致するブロックの一覧をダイアログボックスに 表示します。

RepNames = replace_block('f14','Gain','Integrator')

つぎのコマンドは、clutch システム内の Unlocked サブシステムにある Gain が 'bv' のすべてのブロックを Integrator ブロックで置き換えます。Simulink は、置換を 行う前に、一致するブロックの一覧をダイアログボックスに表示します。

replace_block('clutch/Unlocked','Gain','bv','Integrator')

つぎのコマンドは、f14 システム内の Gain ブロックを Integrator ブロックで置き 換えますが、ダイアログボックスを表示しません。

replace_block('f14','Gain','Integrator','noprompt')

参考 find_system, set_param

目的 Simulink システムを保存します。 表示 save_system save_system('sys') save_system('sys', 'newname') save_system は、カレントの最上位システムをそのカレントの名前のファイルに保 詳細 存します。 save_system('sys')は、指定した最上位システムをそのカレントの名前のファイルに 保存します。システムは開いていなければなりません。 save_system('sys', 'newname')は、指定した最上位システムを指定した新しい名前の ファイルに保存します。システムは開いていなければなりません。 例題 つぎのコマンドは、カレントのシステムを保存します。 save_system つぎのコマンドは、vdp システムを保存します。 save_system('vdp') つぎのコマンドは、vdp システムを 'myvdp' という名前のファイルに保存します。 save_system('vdp', 'myvdp') 参考 close_system, new_system, open_system

目的 Simulink のシステムパラメータとブロックパラメータを設定します。

表示 set_param('obj', 'parameter1', value1, 'parameter2', value2, ...)

詳細 set_param('obj', 'parameter1', value1, 'parameter2', value2, ...) は、'obj' がシステムまたはブロックのパス名であるとき、指定したパラメータを指定した値に設定します。パラメータ名については大文字と小文字の区別はありません。値の文字列については大文字と小文字は区別されます。ダイアログボックスの項目に対応するパラメータはすべて文字列の値をもっています。モデルパラメータとブロックパラメータは、付録 A にリストしてあります。

シミュレーション中にワークスペース上のブロックパラメータ値を変更し、その 変更に従ってブロック線図を更新することも可能です。この操作を行うには、コ マンドウィンドウ上で変更を行った後、そのモデルウィンドウをアクティブにし て、Edit メニューから Update Diagram を選択してください。

注意 ほとんどのブロックパラメータの値は文字列として指定しなければなりま せん。これに対する2つの例外は、すべてのブロックに共通の Position と UserData のパラメータです。

例題

つぎのコマンドは、vdp システムの Solver および StopTime パラメータを設定します。

set_param('vdp', 'Solver', 'ode15s', 'StopTime', '3000')

つぎのコマンドは、vdp システム内のブロック Mu の Gain を 1000(スティッフ) に設定します。

set_param('vdp/Mu', 'Gain', '1000')

つぎのコマンドは、vdp システム内の Fcn ブロックの位置を設定します。

set_param('vdp/Fcn', 'Position', [50 100 110 120])

つぎのコマンドは、mymodel システム内の Zero-Pole ブロックに対する Zeros お よび Poles パラメータを設定します。

set_param('mymodel/Zero-Pole', 'Zeros', '[2 4]', 'Poles', '[1 2 3]')

つぎのコマンドは、マスクされたサブシステム内にあるブロックに対する Gain パラメータを設定します。変数 k が、Gain パラメータと関連付けられています。 set_param('mymodel/Subsystem', 'k', '10')

つぎのコマンドは、システム mymodel 内の Compute という名前のブロックの OpenFcn コールバックパラメータを設定します。関数 'my_open_fcn' は、ユーザ が Compute ブロックをダブルクリックすると実行されます。詳細については、4-71 ページの "コールバックルーチンの使用法"を参照してください。

set_param('mymodel/Compute', 'OpenFcn', 'my_open_fcn')

get_param, find_system

目的 Simulink ブロックライブラリを開きます。

表示 simulink

詳細 Microsoft Windows では、simulink コマンドは Simulink ブロックライブラリブラ ウザを開きます(またはアクティブにします)。UNIX では、Simulink ライブラ リウィンドウを開きます。

simulink

11

Simulink デバッガ

デバッガの起動 ・・・・・・・・・・・・・・・	•	•	•	•	.11-3
シミュレーションの開始............			•		.11-4
デバッガのコマンドラインインタフェースを使って	•		•		.11-6
オンラインヘルプ・・・・・・・・・・・・・・			•		.11-7
シミュレーションの実行.............			•		.11-8
プレークポイントの設定.............					11-11
シミュレーションに関する情報の表示			•		11-16
モデルに関する情報の表示・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・			•		11-20
デバッガコマンドリファレンス........					11-24

Simulink デバッガは、Simulink モデルに含まれるバグを検出し、その原因分析を 行うツールです。デバッガを使うことで、シミュレーションを段階的に実行させ て、ブロックの状態、入力、出力を表示できます。Simulink デバッガは、グラ フィカルユーザインタフェースとコマンドラインインタフェースの双方をもって います。コマンドラインインタフェースにより、すべてのデバッガの機能にアク セスすることができます。グラフィカルインタフェースにより、デバッガの最も 一般的に利用される機能にアクセスすることができます。どちらのインタフェー スを利用することもできますが、ドキュメンテーションではまずグラフィカルイ ンタフェースの使用法を示し、そのつぎにコマンドラインインタフェースを使っ た作業について説明します。

デバッガの起動

デバッガを起動するには、デバッグしたいモデルをオープンして、Tools メ ニューから Debugger を選択します。デバッガウィンドウはつぎのように表示さ れます。



sldebug コマンド、またはデバッガコントロールのもとでモデルを起動するため に sim コマンドのデバッグオプションを使ってコマンドラインからデバッガを起 動することもできます (sim オプションの詳細は、5-38 ページの sim を参照) たと えば、つぎのコマンド

sim('vdp',[0,10],simset('debug','on'))

または

sldebug 'vdp'

は、Simulink デモモデル vdp をメモリにロードし、シミュレーションを開始し、 モデルの実行リストの最初のブロックにおいてシミュレーションを停止します。

注意 GUIモードでデバッガを起動するときは、明示的にシミュレーションを起 動しなければなりません。詳細は、11-4ページの"シミュレーションの開始"を参 照してください。

シミュレーションの開始

シミュレーションを開始するには、デバッガのツールバーで Start/Continue ボタンを選択します。



シミュレーションは開始され、実行される最初のブロックで停止します。デバッ ガはモデルウィンドウのブラウザパネルをオープンし、モデルの実行が停止され るブロックを強調します。



デバッガは、コマンドラインモードで実行しているときは、シミュレーションの 開始時間とデバッグコマンドプロンプトを MATLAB コマンドウィンドウに表示 し、デバッガが GUI モードで実行しているときはデバッガの出力パネルに表示 します。

Bandak dabaggat selp		- 10 B
00.04 P # 40.00 01	*	CCome -
Multi e ponét	Outrial Kenudar	these sheat seen
Blacks (49)(4)	124 + 1) exponent of source index, endor
	and the set	00.31
Manager watch joint Broat an constituent	1	
 Des contemps Des contemps Des contemps Monstrain dest 		
T' trafi values Disc et time .	4	

コマンドプロンプトは、ブロックインデックス(11-6ページの"ブロックイン デックスについて"を参照)と実行される最初のブロック名を表示します。

注意 デバッガを GUI モードで開始する場合、デバッガのコマンドラインインタ フェースは、MATLAB コマンドウィンドウでもアクティブです。しかし、グラ フィカルインタフェースとコマンドラインインタフェース間の同期エラーを防ぐ ために、コマンドラインインタフェースの使用は避けてください。

この点で、ブレークポイントを設定し、シミュレーションをステップ単位で実行 し、つぎのブレークポイントまたは最後までシミュレーションを続け、データを 調べ、その他のデバッグ作業を行うことができます。つぎの節では、デバッガの グラフィカルコントロールを使ってこれらのデバッグ作業を行う方法について説 明します。

デバッガのコマンドラインインタフェースを使って

コマンドラインモードでは、MATLAB コマンドウィンドウのデバッガコマンド ラインにコマンドを入力して、デバッガをコントロールします。デバッガは、デ バッガコマンドの省略形を受け入れます。コマンドの省略形と繰り返し可能なコ マンドのリストは、11-24 ページの"デバッガコマンドリファレンス"を参照して ください。MATLAB コマンドラインで空のコマンド(Return キーを押す)を入力 することによってコマンドを繰り返すことができます。

ブロックインデックスについて

多くの Simulink デバッガコマンドとメッセージは、ブロックを参照するために ブロックインデックスを利用します。ブロックインデックスは、s:b の形式で、s はデバッグされるモデル内のシステムを識別する整数で、b はシステム内のプ ロックを識別する整数です。たとえば、ブロックインデックス 0:1 は、モデルの 0 システムのブロック 1 を参照します。slist コマンドは、デバッグされるモデル 内の各プロックに対するプロックインデックスを表示します。(11-43 ページの slist を参照)。

MATLAB ワークスペースへのアクセス

sldebug プロンプトでは任意の MATLAB 表現をタイプすることができます。た とえば、ブレークポイントにおいてモデルの時間と出力を tout および yout とし てロギングしていると仮定します。そのとき、つぎのコマンド

(sldebug ...) plot(tout, yout)

は、プロットを作成します。sldebug コマンドの完全なまたは省略した名前と同 じ名前をもつ変数にアクセスしたいと仮定します。たとえば、s は step コマンド の省略形です。sldebug プロンプトで s とタイプすると、モデルを進めます。し かし、

(sldebug...) eval('s')

は、変数 s の値を表示します。

オンラインヘルプ

デバッガのツールバーの Help ボタンを選択するか、テキストカーソルがデバッ ガパネル内またはテキストフィールドにある場合は F1 キーを押すことによって、 デバッガのオンラインヘルプを表示することができます。Help ボタンを押すと、



MATLAB ヘルプブラウザにデバッガのヘルプを表示します。F1 キーを押すと、 デバッガパネルまたはカレントでキーボード入力がフォーカスされているテキス トフィールドに対するヘルプを表示します。コマンドラインモードでは、デバッ グプロンプトで help とタイプすることにより、デバッガコマンドの簡単な内容 を表示することができます。

シミュレーションの実行

Simulink デバッガを使って、カレントでつぎのようにサスペンドされている点か らシミュレーションを実行することができます。

- シミュレーションの終了点
- つぎのブレークポイント (11-11 ページの"ブレークポイントの設定"を参照)
- つぎのブロック
- つぎの時間ステップ

GUI モードでは、デバッガツールバーの適切なボタンを選択することによって、 進める量を選択します。

Next BlockNext Time Step



または、コマンドラインモードでは、適切なデバッガコマンドを入力します。

コマンド	シミュレーションの進行状況
step	1 ブロック進行させます
next	1時間ステップ進行させます
continue	つぎのブレークポイント進行させます
run	ブレークポイントを無視して、シミュレーションの終了点ま で進行させます

シミュレーションの継続

GUI モードでは、デバッガは、シミュレーションが何らかの理由でサスペンド されたときに Run/Continue を赤色にします。シミュレーションを続行するには、 Run/Continue ボタンを選択します。コマンドラインモードでは、continue を入力 してシミュレーションを続行します。デバッガは、つぎのブレークポイントまで (11-11ページの"ブレークポイントの設定")またはシミュレーションの終了点の いずれか最初にあるほうまでシミュレーションを続行します。

シミュレーションをノンストップで実行

run コマンドを使って、シミュレーションのカレントの点から終了点まで間にあ るブレークポイントを飛ばしてプログラムを実行することができます。シミュ レーションの終了点で、デバッガは MATLAB コマンドラインに戻ります。モデ ルのデバッグを続けるには、デバッガを再起動しなければなりません。

ブロック毎に実行

シミュレーションを 1 ブロック進めるには、デバッガツールバーで 🏴 をクリッ クするか、デバッガがコマンドラインモードで実行している場合は、デバッガプ ロンプトに step を入力します。デバッガはカレントブロックを実行し、停止し、 モデルのブロックの実行の順番でのつぎのブロックを強調表示します(11-20 ページの"モデルのブロック実行順序の表示"を参照)。たとえば、つぎの図は、 モデルの最初のブロックの実行後の vdp ブロック線図を示します。



実行するつぎのブロックがサブシステムブロックにある場合は、デバッガはサブ システムのブロック線図をオープンし、つぎのブロックを強調表示します。

ブロックを実行した後で、デバッガはブロックの入力 (U) と出力 (Y) を印刷し、 デバッガ出力パネルにデバッグコマンドプロンプトを再表示するか (GUI モード) または MATLAB コマンドウィンドウ (コマンドラインモード) に再表示しま す。デバッガプロンプトは実行するつぎのブロックを表示します。

(sldebug @0:0 'vdp/Integrator1'): step

Y1 = [2] (sldebug @0:1 'vdp/Out1'):

時間ステップの境界を超える

モデルのブロック実行リストの最後のブロックを実行した後で、デバッガはシ ミュレーションをつぎの時間ステップに進行させ、シミュレーションを停止しま す。時間ステップの境界を超えたことをユーザに通知するため、デバッガは GUI モードではデバッガ出力パネルに現在の時間を表示し、コマンドライン モードでは MATLAB コマンドウィンドウに表示します。たとえば、vdp モデル の最初の時間ステップの最後のブロックを実行し終えると、デバッガ出力パネル または MATLAB コマンドウィンドウではつぎの出力結果になります。

(sldebug @0:8 'vdp/Sum'): step

U1 = [2]

U2 = [0]

Y1 = [-2]

[Tm=0.0001004754572603832] **Start** of system 'vdp' outputs

マイナー時間ステップづつ進行

メジャー時間ステップと同様に、マイナー時間ステップ内のでブロック毎に実行 することができます。マイナー時間ステップでブロック毎に実行するには、デ バッガの Break on conditions パネルの Minor time steps オプションをチェックす るか、デバッガコマンドプロンプトで minor を入力します。

時間ステップ毎に実行

時間ステップ毎に実行するには、
ご をクリックするか、デバッガコマンドラインで next コマンドを入力します。デバッガは現在の時間ステップの残りのブロックを実行し、つぎの時間ステップの開始点へとシミュレーションを進めます。たとえば、デバッグモードで vdp モデルの開始後に next を入力すると、MATLABコマンドウィンドウにつぎのメッセージが表示されます。

[Tm=0.0001004754572603832] **Start** of system 'vdp' outputs

ブレークポイントの設定

Simulink デバッガを使って、ブレークポイントと呼ばれるシミュレーション中の 停止点を定義することができます。それからブレークポイント間でデバッガの continue コマンドを使ってシミュレーションをブレークポイント毎に実行するこ とができます。デバッガは2つのタイプ、無条件ブレークポイントと条件付きブ レークポイントのブレークポイントを定義することができます。無条件ブレーク ポイントは、シミュレーションが前もって指定したブロックまたは時間ステップ に到達したときに生じます。条件付きブレークポイントは、前もって指定した条 件がシミュレーションで生じたときに発生します。

プログラムのある特定のポイントで問題が生じることがわかっているときや、あ る条件が生じたときに問題が起こることをわかっているときに、ブレークポイン トは何かと有効に利用することができます。適切なブレークポイントを定義し、 continue コマンドによってシミュレーションを実行することにより、シミュレー ション上の問題が生じているポイントまで直ちにスキップすることができます。

デバッガツールバーのブレークポイントボタンをクリックすることにより、ブレークポイントを設定します。



または適切なブレークポイント条件をチェックします (GUI モード)

Ð	exkonconditions
с	Zero crossings
c	Stop size limited by state
Г	Minor time steps
r	NaN values
Et	ap at time .

または適切なブレークポイントコマンドを入力します(コマンドラインモード)。

コマンド	シミュレーションの停止条件
break <gcb s:b="" =""></gcb>	ブロックの実行開始時
bafter <gcb s:b="" =""></gcb>	ブロックの実行終了時

コマンド	シミュレーションの停止条件
tbreak [t]	シミュレーション時間ステップ
nanbreak	アンダーフローまたはオーバフロー (NaN) また は無限大 (Inf) の値が生じたとき
xbreak	シミュレーションステップサイズを決定する状態 にシミュレーションが到達したとき
zcbreak	シミュレーション時間ステップ間でゼロクロッシ ングが発生したとき

ブロックでブレークポイントを設定

デバッガを使ってブロックの実行開始時や、ブロックの実行終了時にブレークポイントを指定することができます(コマンドラインモードのみ)。

ブロックの実行開始時にブレークポイントを指定

ブロックの実行開始時にブレークポイントを設定すると、デバッガは各時間ス テップでのブロックに到達するときにシミュレーションを停止します。ブレーク ポイントをグラフィカルに設定するか、コマンドラインモードでブロックイン デックスによってブロックを指定することができます。ブロックの実行の開始時 にグラフィカルにブレークポイントを設定するには、モデルウィンドウでブロッ クを選択し、 T をデバッガのツールバーでクリックするか、

break gcb

をデバッガコマンドラインで入力します。ブロックインデックスを使ってブロックを指定するには(コマンドラインモードのみ)、以下を入力します。

break s:b

ここで、s:b はブロックのインデックスです(11-6ページの"ブロックインデック スについて"を参照)。 注意 バーチャルブロックにおいてブレークポイントは設定できません。バー チャルブロックは、関数が純粋にグラフィカルであるブロックです。これは、モ デルの計算ブロックでのグループ分けや関係を示します。デバッガは、バーチャ ルブロックでブレークポイントを設定しようとしたときは、ワーニングを表示し ます。slist コマンドを使ってモデルの非バーチャルブロックのリストを取得でき ます (11-21 ページの"モデルの非バーチャルブロックの表示"を参照)。

GUI モードでは、デバッガの Watch points パネルは、ブレークポイントが存在 するブロックを表示します。

3	locks	<0.0
rip/Das		PC
	. manage	transfer by man let

ブロックの実行終了時にブレークポイントを指定

コマンドラインモードでは、デバッガは bafter コマンドを使ってブロックの実行 の終了時にブレークポイントを設定することができます。break を使って、ブ ロックをグラフィカルに、またはブロックインデックスによって指定することが できます。

ブレークポイントをブロックからクリア

ブレークポイントを一時的にクリアするには、Watch points パネル (GUI モードのみ)のブレークポイントの隣りの最初のチェックボックスのチェックを外します。GUI モードでブレークポイントを恒久的にクリアするには、Watch points パネルのブレークポイントを選択し、Remove watch point ボタンをクリックします。コマンドラインモードでは、clear コマンドを使ってブレークポイントをクリアします。ブロックインデックスを入力したり、モデル線図のブロックを選択して clear コマンドの引数として gcb を入力することによって、ブロックを指定することができます。

時間ステップでブレークポイントを設定

時間ステップでブレークポイントを設定するには、デバッガの Stop at time フィールドで時間を入力するか (GUI モード)、または tbreak コマンドを使って 時間を入力します。デバッガは、指定した時間の後の最初の時間ステップの開始 時にシミュレーションを停止します。たとえば、デバッグモードで vdp を起動 し、つぎのコマンドを入力します。

tbreak 9 continue

は、continue コマンドの出力によって示されたように時間ステップ 9.0785 の開始 時にシミュレーションを停止します。

[Tm=9.07847133212036] **Start** of system 'vdp' outputs

非有限値で停止

デバッガの NaN values オプションをチェックするか、nanbreak コマンドを入力 することにより、計算値が無限大かシミュレーションを実行するマシンによって 表現可能な値の範囲外であるときに、シミュレーションを停止させます。このオ プションは、Simulink モデルの計算エラーを正確に指摘するのに役立ちます。

ステップサイズを制限したステップで停止

Step size limited by state オプションをチェックするか xbreak コマンドを入力する と、デバッガはモデルが可変ステップソルバを利用して、ソルバがとりうるス テップサイズを制限する状態があるときに、シミュレーションを停止します。こ のコマンドは、多くの解くべきシミュレーション時間ステップ数を必要とするモ デルのデバッグに役立ちます。

ゼロクロッシングにおいて停止

Zero crossings オプションをチェックするか、zcbreak コマンドを入力すると、ゼロクロッシングが生じるブロックを含むモデル内でサンプリングされていないゼロクロッシングを検出したときに、シミュレーションを停止します。停止した後で、Simulink はゼロクロッシングのモデル内の位置、時間、タイプ (rising または falling)を出力します。たとえば、zeroxing デモモデルの実行の開始時にゼロクロッシングブレークポイントを設定します。

sldebug zeroxing

[Tm=0] **Start** of system 'zeroxing' outputs (sldebug @0:0 'zeroxing/Sine Wave'): zcbreak Break at zero crossing events is enabled.

そしてシミュレーションを続行します。

(sldebug @0:0 'zeroxing/Sine Wave'): continue

は、以下における rising タイプのゼロクロッシングになります。

[Tm=0.34350110879329] Breaking at block 0:2

[Tm=0.34350110879329] Rising zero crossing on 3rd zcsignal in block 0:2 'zeroxing/ Saturation'

モデルがサンプリングされていないゼロクロッシングを生成可能なブロックを含 まない場合、コマンドはこの事実を通知するメッセージを表示します。

シミュレーションに関する情報の表示

Simulink デバッガは、ブロックの状態、ブロックの入力と出力、モデルの実行中の情報を表示するコマンドを提供します。

ブロック I/O の表示

デバッガを使ってデバッガツールバーの適切なボタンを選択することにより、ブロック I/O を表示することができます。



または、適切なデバッガコマンドを入力することによっても可能です。

コマンド	ブロックの I/O をいつ表示するか
probe	即時表示します
disp	ブレークポイント毎に表示します
trace	ブロックの実行時に表示します

選択したブロックの I/O を表示

ブロックの I/O を表示するには、ブロックを選択し、GUI モードでは MP をクリックするか、コマンドラインモードでは probe コマンドを入力します。

コマンド	説明
probe	probe モードに入るか、または抜けます。probe モードでは、 デバッガはモデルのブロック線図で選択した任意のブロック のカレントの入力と出力を表示します。任意のコマンドをタ イプすると、デバッガは probe モードから抜けます。

コマンド	説明
probe gcb	選択したブロックの I/O を表示します
probe s:b	システム番号 s およびブロック番号 b で指定されたブロック の I/O を表示します。

デバッガは、デバッガ出力パネル (GUI モード) または MATLAB コマンドウィ ンドウに選択したブロックのカレントの入力および出力を表示します。

probe コマンドは、I/O が表示されないブロックの I/O を調べる必要があるときに 役立ちます。たとえば、ブロック単位でモデルを実行するコマンド step を使用す るとします。モデルが段階的に実行される度に、デバッガはカレントプロックの 入出力を表示します。probe コマンドを使って、他のプロックの I/O も調べるこ とができます。同様に、時間ステップ単位で、モデルを段階的に実行させるため に、next コマンドを使用したと仮定します。next コマンドは、プロック I/O を表 示しません。しかし、next コマンドを入力した後でプロックの I/O を調べる必要 がある場合は、probe コマンドを使って行うことができます。

ブロック I/O をブレークポイントで自動的に表示

disp コマンドは、シミュレーションを停止したときに指定したブロックの入出力 を表示します。ブロックインデックスを入力するか、ブロック線図内を選択し て、disp コマンド引数として gcb を入力することによって、ブロックを指定する ことができます。undisp コマンドを使って、表示点のデバッガのリストからブ ロックを削除することができます。たとえば、block 0:0 を削除するには、モデル 線図内でブロックを選択して undisp gcb を入力するか、単に undisp 0:0 を入力し ます。

注意 ブレークポイントでのブロック I/O の自動表示は、デバッガの GUI モード では利用できません。

disp コマンドは、シミュレーションを段階的に実行するときに特定のブロックや 複数のブロックの I/O をモニタする必要があるときに役立ちます。disp コマンド を使って、モニタしたいブロックを指定することができ、デバッガはその後各ス テップにおいてそれらのブロックの I/O を再表示します。デバッガは、step コマ ンドを使ってブロック毎にモデルを実行するときには、カレントのブロックの I/ O を常に表示します。そのため、カレントのプロックの I/O のみを監視すること に興味がある場合には、disp コマンドを使う必要がありません。

ブロック I/O の監視

ブロックを監視するには、ブロックを選択して、デバッガツールバーで Part をク リックするか、trace コマンドを入力します。GUI モードでは、ブレークポイン トがブロックに存在する場合は、Watch points パネルでブロックに対する watch チェックボックスをチェックすることによって、監視を設定することができま す。コマンドラインモードでは、trace コマンドでブロックインデックスを指定 することにより、ブロックを指定することもできます。untrace コマンドを使っ て、トレースポイントのデバッガのリストからブロックを削除することができま す。

デバッガは、ブロックが実行するときに監視されているブロックの I/O を表示します。ブロックを監視することによって、シミュレーションを停止せずにブロックの I/O の完全な記録を得ることができます。

代数ループの情報を表示

atrace コマンドを使って、モデルの代数ループが解かれるたびに代数ループに関する情報をデバッガに表示します(3-18ページの"代数ループ"を参照)。コマンドは、表示する情報量を指定する単一の引数をもちます。

コマンド	各代数ループをどのように表示するか
atrace 0	情報を表示しません
atrace 1	ループの可変解、ループを解くのに要する繰り返し回数、推 定された解の誤差を表示します
atrace 2	レベル1と同等の情報を表示します
atrace 3	レベル2の情報に加え、ループの解法に用いるヤコビアン行 列に関する情報を表示します
atrace 4	レベル3の情報に加え、ループ変数の中間解に関する情報を 表示します

システムの状態の表示

states デバッグコマンドは、MATLAB コマンドウィンドウにシステムの状態のカレント値をリスト表示します。たとえば、つぎのコマンドは、Simulinkの跳ねるボールのデモ (bounce)の1番目と2番目の時間ステップでの実行に関する状態を表示したものです。
sldebug bounce [Tm=0] **Start** of system 'bounce' outputs (sldebug @0:0 'bounce/Position'): states Continuous state vector (value, index, name): 10 0 (0:0 'bounce/Position') 15 1 (0:5 'bounce/Velocity') (sldebug @0:0 'bounce/Position'): next [Tm=0.01] **Start** of system 'bounce' outputs (sldebug @0:0 'bounce/Position'): states Continuous state vector (value, index, name): 10.1495095 0 (0:0 'bounce/Position') 14.9019 1 (0:5 'bounce/Velocity')

積分情報の表示

ishow コマンドは、積分情報の表示を切り替えます。使用可能なときは、このオ プションにより、デバッガは時間ステップ毎に、または、時間ステップのサイズ を制限する状態に遭遇する毎にメッセージを表示します。最初の場合では、デ バッガは時間ステップのサイズを、つぎのように表示します。

[Tm=9.996264188473381] Step of 0.01 was taken by integrator

2番目の場合では、デバッガは時間ステップのサイズをカレントに決定する状態 を、つぎのように表示します。

[Ts=9.676264188473388] Integration limited by 1st state of block 0:0 'bounce/Position'

モデルに関する情報の表示

シミュレーションに関する情報を提供するのに加えて、デバッガは、シミュレー ションを構成するモデルに関する情報も提供します。

モデルのブロック実行順序の表示

Simulink は、モデルの初期化中に、シミュレーションの実行の開始時にブロック を実行する順番を決定します。シミュレーション中に、Simulink は実行順に並べ られたプロックのリストを保持します。このリストは、"ソートされたリスト" と呼ばれます。GUI モードでは、デバッガは Execution Order パネルにソートさ れたリストを表示します。コマンドラインモードでは、slist コマンドはモデルの プロック実行順を MATLAB コマンドウィンドウに表示します。リストには、各 コマンドに対するブロックインデックスが含まれます。

---- Sorted list for 'vdp' [12 blocks, 9 nonvirtual blocks, directFeed=0]

- 0:0 'vdp/Integrator1' (Integrator)
- 0:1 'vdp/Out1' (Outport)
- 0:2 'vdp/Integrator2' (Integrator)
- 0:3 'vdp/Out2' (Outport)
- 0:4 'vdp/Fcn' (Fcn)
- 0:5 'vdp/Product' (Product)
- 0:6 'vdp/Mu' (Gain)
- 0:7 'vdp/Scope' (Scope)
- 0:8 'vdp/Sum' (Sum)

ブロックの表示

モデルのブロック線図のどのブロックが特定のインデックスに対応するかを確定 するには、コマンドプロンプトで bshow s:b と入力してください。ここで、s:b は ブロックインデックスです。bshow コマンドは、(必要ならば)ブロックを含む システムをオープンし、システムウィンドウ内でブロックを選択します。

モデルの非バーチャルブロックの表示

systems コマンドは、デバッグ対象となるモデル内の非バーチャルシステムのリ ストを印刷します。たとえば、Simulink clutch デモ (clutch) にはつぎのシステム が含まれます。

sldebug clutch

[Tm=0] **Start** of system 'clutch' outputs (sldebug @0:0 'clutch/Clutch Pedal'): systems

- 0 'clutch'
- 1 'clutch/Locked'
- 2 'clutch/Unlocked'

注意 systems コマンドは、モデル内に存在する純粋にグラフィカルなサブシステ ムをリスト表示しません。純粋にグラフィカルなシステムとは、モデルのブロッ ク線図はサプシステムプロックとして表示され、Simulink が親システムの一部と して解を求めるシステムのことです。Simulink モデルでは、ルートシステムとト リガ付きまたはイネーブルシステムが真のシステムと考えられます。それ以外の システムは、すべてバーチャル(すなわち、グラフィカル)で、そのため systems コマンドによって生成されるリストには表示されません。

モデルの非バーチャルプロックの表示

slist コマンドは、モデル内の非バーチャルブロックのリストを表示します。リス ト表示は、ブロックをシステムによってグループ分けします。たとえば、つぎの コマンドは、Van der Pol (vdp) デモモデル内の非バーチャルブロックのリストを 生成します。

sldebug vdp

[Tm=0] **Start** of system 'vdp' outputs

(sldebug @0:0 'vdp/Integrator1'): slist

- ---- Sorted list for 'vdp' [12 blocks, 9 nonvirtual blocks, directFeed=0]
- 0:0 'vdp/Integrator1' (Integrator)
- 0:1 'vdp/Out1' (Outport)
- 0:2 'vdp/Integrator2' (Integrator)
- 0:3 'vdp/Out2' (Outport)
- 0:4 'vdp/Fcn' (Fcn)
- 0:5 'vdp/Product' (Product)
- 0:6 'vdp/Mu' (Gain)
- 0:7 'vdp/Scope' (Scope)
- 0:8 'vdp/Sum' (Sum)

注意 slist コマンドは、純粋にグラフィカルなシステムをリスト表示しません。 純粋にグラフィカルなブロックとは、計算ブロック間の関係やグループ分けを示 すブロックのことです。

潜在的なゼロクロッシングをもつブロックの表示

zclist は、シミュレーション中にサンプリングされていないゼロクロッシングが 生じたブロックのリストを出力します。たとえば、zclist は、clutch サンプルモデ ルに対しては、以下のようなりストを出力します。

(sldebug @0:0 'clutch/Clutch Pedal'): zclist

- 2:3 'clutch/Unlocked/Sign' (Signum)
- 0:4 'clutch/Lockup Detection/Velocities Match' (HitCross)
- 0:10 'clutch/Lockup Detection/Required Friction for Lockup/Abs' (Abs)
- 0:11 'clutch/Lockup Detection/Required Friction for Lockup/ Relational Operator' (RelationalOperator)
- 0:18 'clutch/Break Apart Detection/Abs' (Abs)
- 0:20 'clutch/Break Apart Detection/Relational Operator' (RelationalOperator)
- 0:24 'clutch/Unlocked' (SubSystem)
- 0:27 'clutch/Locked' (SubSystem)

代数ループの表示

ashow コマンドは、指定した代数ループまたは指定したブロックを含む代数ルー プを強調表示します。指定した代数ループを強調するには、ashow s#n, とタイプ します。ここで、s はループを含むシステムのインデックスで(11-20 ページの" モデルのブロック実行順序の表示"を参照)、n はシステム内のループのインデッ クスです。現在選択されているブロックを含むループを表示するには、ashow gcb を入力します。指定したブロックを含むループを表示するには、ashow s:b と 入力します。ここで、s:b はブロックのインデックスです。モデルのブロック線 図から代数ループの強調表示を消すには、ashow clear と入力してください。

デバッガのステータスの表示

GUI モードでは、デバッガは条件付きブレークポイントのような様々なデバッ グのオプションの設定を Status パネルに表示します。コマンドラインモードで は、status コマンドはデバッガの設定を表示します。たとえば、つぎのコマンド は、vdp モデルに対する初期デバッグ設定を表示します。

sim('vdp',[0,10],simset('debug','on'))

[Tm=0] **Start** of system 'vdp' outputs

(sldebug @0:0 'vdp/Integrator1'): status

Current simulation time: 0 (MajorTimeStep)

Last command: ""

Stop in minor times steps is disabled.

Break at zero crossing events is disabled. Break when step size is limiting by a state is disabled. Break on non-finite (NaN,Inf) values is disabled. Display of integration information is disabled. Algebraic loop tracing level is at 0.

デバッガコマンドリファレンス

つぎの表は、デバッガコマンドを表示したものです。この表の Repeat 欄は、コマンドラインで Return キーを押すことにより、コマンドを繰り返し実行できるかどうかを示しています。コマンドの詳細な説明については、表を参照してください。

コマンド	省略 形	繰り返 し	詳細
ashow	as	なし	代数ループを表示します
atrace	at	なし	代数ループのトレースレベルを設定しま す
bafter	ba	なし	ブロックの実行後にブレークポイントを 挿入します
break	b	なし	ブロックの実行前にブレークポイントを 挿入します
bshow	bs	なし	指定したブロックを表示します
clear	cl	なし	ブロックからブレークポイントを除去し ます
continue	с	あり	シミュレーションを続行します
disp	d	あり	シミュレーション停止時に、プロック I/O を表示します
help	? or h	なし	デバッガコマンドのヘルプを表示します
ishow	i	なし	積分情報表示のオン / オフを切り替えま す
minor	m	なし	マイナーステップモードのオン / オフを 切り替えます
nanbreak	na	なし	有限でない値にブレークポイントを設定/ 解除します

コマンド	省略 形	繰り返 し	詳細
next	n	あり	つぎの時間ステップの開始時までシミュ レーションを進めます
probe	р	なし	1 つのブロックの I/O を表示します
quit	q	なし	シミュレーションを中断します
run	r	なし	シミュレーションを最後まで実行します
slist	sli	なし	モデルの非バーチャルブロックをリスト 表示します
states	state	なし	カレントの状態値を表示します
status	stat	なし	有効になっているデバッグオプションを 表示します
step	S	あり	つぎのブロックまでシミュレーションを 進めます
stop	sto	なし	シミュレーションを停止します
systems	sys	なし	モデルの非バーチャルシステムをリスト 表示します
tbreak	tb	なし	時間ブレークポイントを設定 / 解除しま す
trace	tr	あり	実行する毎にブロックの I/O を表示しま す
undisp	und	あり	表示ポイントのリストからブロックを削 除します
untrace	unt	あり	トレースポイントのリストからブロック を削除します
xbreak	X	なし	ステップサイズを制限する状態にデバッ ガが遭遇したときにシミュレーションを 停止します

コマンド	省略 形	繰り返 し	詳細
zcbreak	zcb	なし	サンプリングされていないゼロクロッシ ングイベントが検出されたときにシミュ レーションを停止します
zclist	zcl	なし	サンプリングされていないゼロクロッシ ングを伴うブロックをリスト表示します

表示 ashow <gcb | s:b | s#n | clear>

引数	s:b	ブロックのシステムインデックス s、ブロックインデックス b
	gcb	カレントブロック
	s#n	システム s 内の n という番号の代数ループ
	clear	ループに対するカラー表示を解除します
詳細	ashow gcb また します。 ashov ashow clear は	とは ashow s:b は、指定されたブロックを含む代数ループを強調表示 w s#n は、システム s 内の n 番目の代数ループを強調表示します。 、モデル線図からループの強調表示を解除します。

参考 atrace, slist

目的	代数ルー	プのトレー	スレベ	ルを設定し	,ます
		ノのドレー	ヘレン	ルで改たし	ገው እ

表示 atrace level

引数 トレースレベル (0 =なし、4 =すべて) level

atrace コマンドは、シミュレーションに対する代数ループトレースレベルを設定 詳細 します。

コマンド	各代数ループをどのように表示するか
atrace 0	代数ループに関する情報を表示しません。
atrace 1	ループ内の可変解、ループを解くのに必要となる反復処理の 回数、推定される解の誤差を表示します。
atrace 2	レベル1と同等の情報を表示します。
atrace 3	レベル2の情報に加え、ループを解くのに用いるヤコビアン 行列に関する情報を表示します。
atrace 4	レベル3の情報に加え、ループ変数の中間解に関する情報を 表示します。

参考

systems, states

目的	ブロックの実行後にブレークポイントを挿入します。	
表示	bafter gcb bafter s:b	
引数	s:b gcb	ブロックのシステムインデックス s、ブロックインデックス b カレントブロック
詳細	bafter コマン す。	ドは、指定したブロックの実行後にブレークポイントを挿入しま
参考	break, xbreak, tb	oreak, nanbreak, zcbreak, slist

break

目的	ブロックの実行前にブレークポイントを挿入します。	
表示	break gcb break s:b	
引数	s:b gcb	ブロックのシステムインデックス s、ブロックインデックス b カレントブロック
詳細	break コマント	ドは、指定したブロックの実行前にブレークポイントを挿入します。
参考	bafter, tbreak, xł	preak, nanbreak, zcbreak, slist

目的 指定したブロックを表示します。

表示 bshow s:b

引数 s:b ブロックのシステムインデックス s、ブロックインデックス b

詳細 このコマンドは、指定したブロックを含むモデルウィンドウを開き、ブロックを 選択します。

参考 slist

目的	ブロックからブレークポイントを取り除きます。	
表示	clear gcb clear s:b	
引数	s:b gcb	ブロックのシステムインデックス s、ブロックインデックス b カレントプロック
詳細	clear コマンド	は、指定したブロックからブレークポイントを除去します。
参考	bafter, slist	

目的 シミュレーションを続行します。

表示 continue

詳細 continue コマンドは、カレントのブレークポイントからシミュレーションを続行 します。シミュレーションは、他のブレークポイントに到達するか、最終時間ス テップに到達するまで続行します。

参考 run, stop, quit

目的 シミュレーションが停止する時に、ブロックの I/O を表示します。

表示	disp gcb
	disp s:b
	disp

引数 s:b ブロックのシステムインデックス s、ブロックインデックス b gcb カレントブロック

詳細 disp コマンドは、ある1つのブロックを表示ポイントとして登録します。デバッガは、シミュレーションが停止すると、常に MATLAB コマンドウィンドウにすべての表示ポイントの入出力を表示します。引数なしの disp コマンドは、表示ポイントをリスト表示します。表示ポイントからブロックを取り除くためには、undisp を使用してください。

参考 undisp, slist, probe, trace

目的 デバッガコマンドのヘルプを表示します。

表示 help

詳細 help コマンドは、コマンドウィンドウにデバッガコマンドのリストを表示します。リスト表示には、それぞれのコマンドの表記法と省略形も表示されます。

目的 積分情報表示のオン / オフを切り替えます。

表示 ishow

詳細 ishow コマンドは、シミュレーション中に積分情報の表示の切り替えスイッチと なります。

参考 atrace

目的 マイナーステップモードのオン / オフを切り替えます。

表示 minor

step

詳細 minor コマンドは、マイナーステップモードに入るか、マイナーステップモード から抜け出すかのいずれかを行うことができます。マイナーステップモードで は、step コマンドはマイナーステップ内でブロック毎にシミュレーションを進め ていきます。マイナーステップモードは、モデルのソートされたブロックリスト の最後のブロックが実行された後、カレントのメジャー時間ステップにまだマイ ナー時間ステップが残されている場合には、step コマンドは、つぎのマイナー時間ステップへとシミュレーションを進行させます。そうでない場合は、step コマ ンドは、つぎのメジャー時間ステップの最初のマイナー時間ステップへとシミュ レーションを進行させます。

参考

目的	有限でない値の時にシミュレーションを停止するモードの設定 / 解除を行いま す。
表示	nanbreak
詳細	nanbreak コマンドは、シミュレーションが有限でない値 (NaN または Inf) に遭遇 したときに、常にシミュレーションを停止するようにデバッガを設定します。有 限でない値のとき、シミュレーションを停止するモードが設定されている場合、 nanbreak によってそのモードを解除することができます。
参考	break, bafter, xbreak, tbreak, zcbreak

目的 つぎの時間ステップの開始時にシミュレーションを進めます。

表示 next

詳細 next コマンドは、カレント時間ステップで評価されていないすべてのブロックを 評価し、つぎの時間ステップの開始時にシミュレーションを停止します。next コ マンドを実行した後、デバッガはつぎの時間ステップで最初に評価されるブロッ クを強調表示し、つぎのステップの時間を表示します。

参考

step

probe

引数

目的 ブロックの状態を表示します。

表示 probe [<s:b | gcb>] [level io | (all)]

s:b ブロックのシステムインデックス s、ブロックインデックス b

gcb カレントブロック

level io ブロックの I/O を表示

level all 入出力、状態、ゼロクロッシングを含むブロックのカレントの状態に関するすべての情報を表示

詳細 probe により、デバッガは probe モードの設定 / 解除を切り替えることができます。probe モードでは、デバッガは選択した任意のブロックの I/O を表示します。probe モードを解除するには、任意のコマンドを入力してください。probe gcb は、カレントで選択されているブロックの I/O を表示します。probe s:b は、インデックスが s:b であるプロックの I/O を表示します。

参考 disp, trace

目的 シミュレーションを中断します。

表示 quit

詳細 quit コマンドは、カレントのシミュレーションを中断します。

参考 stop

目的 シミュレーションを完全に最後まで実行します。

表示 run

詳細 run コマンドは、シミュレーションをカレントのブレークポイントから最後の時 間ステップまで実行します。run コマンドは、ブレークポイントや表示ポイント を無視します。

参考 continue, stop, quit

目的 モデル内の非バーチャルブロックをリスト表示します。

表示 slist

詳細 slist コマンドは、デバッグ対象のモデル内の非バーチャルブロックをリスト表示 します。リストには、リストアップされた各ブロックのブロックインデックスと 名前も表示されます。

参考 systems

目的 カレントの状態値を表示します。

表示 states

詳細 states コマンドは、モデルのカレント状態のリストを表示します。各状態の値、 インデックス、名前が表示されます。

参考 ishow

目的 モデル内の非バーチャルシステムをリスト表示します。

表示 systems

詳細 systems コマンドは、MATLAB コマンドウィンドウに、モデル内の非バーチャル システムをリスト表示します。

参考 slist

目的 有効なデバッグオプションを表示します。

表示 status

詳細 status コマンドは、有効なデバッグオプションのリストを表示します。

目的 つぎのブロックヘシミュレーションを進めます。

表示 step

詳細 step コマンドは、カレント時間ステップにおいて、つぎの評価対象となるブロックを評価します。step コマンドを実行した後で、デバッガはつぎに評価されるブロックとその出力信号ラインを強調表示します。また、デバッガは、デバッガコマンドラインプロンプトの一部として、つぎのブロックの名前も表示します。

参考

next

目的 シミュレーションを停止します。

表示 stop

詳細 stop コマンドは、シミュレーションを停止します。

参考 continue, run, quit

目的 時間ブレークポイントの設定/除去を行います。

表示 tbreak t

tbreak

詳細 tbreak コマンドは、指定した時間ステップにブレークポイントを設定します。ブレークポイントが既に指定した時間に設定されている場合には、tbreak はそのブレークポイントを除去します。時間を指定しない場合は、tbreak はカレントの時間ステップのブレークポイントの設定/除去を切り替えます。

参考 break, bafter, xbreak, nanbreak, zcbreak

目的	ブロックを実行する毎に、ブロックの I/O を表示します。	
表示	trace gcb trace s:b	
引数	s:b gcb	ブロックのシステムインデックス s、ブロックインデックス b カレントブロック
詳細	trace コマンド デバッガは、 す。	は、ある 1 つのブロックをトレースポイントとして登録します。 この登録されたブロックが実行されるたびに、その I/O を表示しま
参考	disp, probe, untr	ace, slist

目的	デバッガの	デバッガの表示ポイントリストからある1つのブロックを除去します。	
表示	undisp gcb undisp s:b		
引数	s:b gcb	ブロックのシステムインデックス s、ブロックインデックス b カレントブロック	
詳細	undisp コマ り除きます	ンドは、デバッガの表示ポイントのリストから指定したブロックを取 。	
参考	disp, slist		

目的	デバッガのトレースポイントのリストからある1つのブロックを除去します。	
表示	untrace gcb untrace s:b	
引数	s:b gcb	ブロックのシステムインデックス s、ブロックインデックス b カレントブロック
詳細	untrace コマンドは、デバッガのトレースポイントのリストから指定したブロッ クを取り除きます。	
参考	trace, slist	

目的 デバッガがステップサイズを制限する状態に遭遇した時にシミュレーションを停止します。

詳細 xbreak コマンドは、デバッガはソルバが取得するステップサイズを制限する状態 に遭遇したときに、モデルの実行を停止します。xbreak モードが既にオンになっ ているときには、xbreak はモードを解除します。

参考 break, bafter, zcbreak, tbreak, nanbreak

表示 xbreak

目的 サンプリングされていないゼロクロッシングイベントが生じたときにシミュレー ションを停止するかどうかを切り替えます。

表示 zcbreak

詳細 zcbreak コマンドにより、デバッガは、サンプリングされていないゼロクロッシングイベントが生じたときにシミュレーションを停止します。ゼロクロッシングブレークモードが既にオンになっているときには、zcbreak はそのモードを解除します。

参考 break, bafter, xbreak, tbreak, nanbreak, zclist
目的 サンプリングされていないゼロクロッシングを伴うブロックをリスト表示します。

表示 zclist

詳細 zclist コマンドは、サンプリングされていないゼロクロッシングが生じるブロッ クのリストを出力します。このコマンドは、リストを MATLAB コマンドウィン ドウに表示します。

参考 zcbreak

12

Performance Tools

Simulink Performance Tools オプション	.12-2
Simulink Accelerator	.12-3
機能方法	12-3
Simulink Accelerator の実行方法	12-4
モデル構造の変更の取り扱い	12-5
Accelerator モードの性能向上	12-6
速度の改良を示さないブロック	12-7
Simulink Accelerator を Simulink Debugger で利用	12-8
Simulink Accelerator とのプログラムインタフェース	12-9
性能の比較	12-10
Simulink Accelerator Build プロセスのカスタマイズ	12-10
Simulink Accelerator での S- ファンクションの実行の制御	12-11
Model Differencing Tool	12-13
表示オプション	12-15
モデルの差分をレポート	12-15
プロファイラ	12-17
どのようにしてプロファイラが働くか	12-17
プロファイラを利用可能にする	12-19
シミュレーションプロファイル	12-20
Model Coverage Tool(モデル補償範囲ツール)	12-23
どのようにして Model Coverage Tool が働くか	12-23
Model Coverage Tool の使用	12-23
テストケースの作成と実行	12-24
カバレージレポート	12-26
カバレージの設定ダイアログボックス	12-29
Model Coverage $\exists \forall \forall \flat k \dots \dots$	12-31

Simulink Performance Tools オプション

Simulink Performance Tools 製品は、つぎのツールを提供します。

- Simulink Accelerator
- Model Differencing Tool
- ・プロファイラ
- Model Coverage Tool

注意 これらのツールを使用するためには、Performance Tools オプションをイン ストールしなければなりません。

Simulink Accelerator

Simulink Accelerator は、Simulink に付属する MathWorks のプロダクトであり、 Simulink モデルの実行を高速化します。Accelerator は、Simulink モデルから C コードを自動生成する MathWorks のプロダクトである Real-Time Workshop と、 C コンパイラを利用して、実行可能なファイルを作成します。Simulink Accelerator は、Real-Time Workshop テクノロジを利用しますが、Real-Time Workshop は実行には必要ありません。また、Windows PC に C コンパイラがイ ンストールされていない場合は、MathWorks が提供する lcc コンパイラを利用で きます。

注意 アクセラレータを使用するためには、Simulink Performance Tools オプショ ンをインストールしなければなりません。

機能方法

Simulink Accelerator は、Simulink が Normal モードのとき(つまり Accelerator モードでないとき)に使用され、インタープリティブなコードに代わりに C コー ドを生成、コンパイルして解析を実行します。Accelerator は、Simulink モデルか ら C コードを生成し、そして MATLAB の MEX 関数がシステムのコンパイラを 呼び出して、生成されたコードをダイナミックに Simulink にリンクさせます。

Simulink Accelerator は、Normal モードで実行するときに Simulink モデルから要 求される計算オーバーヘッドの多くを取り除きます。Simulink Accelerator は、 Simulink で可能な構成を扱うために設計されたブロックを、ユーザ固有のモデル の構成にカスタマイズされたコンパイルバージョンで置き換えることによって機 能します。この方法によって、大規模 Simulink モデル性に対する能を向上させ ることができます。性能向上の度合いは、モデルの規模と複雑さに依存します。 一般的には、組込みの Simulink ブロックを使用したモデルの性能の 2 倍から 6 倍の向上を見込むことができます。

Simulink Accelerator の実行方法

Simulink Accelerator を起動するためには、モデルに対して Simulation メニューから Accelerator を選択します。つぎの図は、F14 制御モデルを使った手順を示します。



図 12-1: Simulink での Accelerator モードの選択

代わりに、ツールバーの右側のプルダウンメニューから Accelerator を選択する こともできます。

シミュレーションを開始するには、Simulation メニューから Start を選択します。 シミュレーションの開始時に、Accelerator は C コードを生成し、コンパイルし ます。その後、Accelerator はつぎのことを行います。

- 生成したコードを modelname_accel_rtw (この場合、f14_accel_rtw)というサブ ディレクトリに置きます。
- カレントの作業ディレクトリにコンパイルした MEX- ファイルを置きます。
- コンパイルしたモデルを実行します。

注意 コードがコンパイルされない場合、最も考えられる理由は、mex コマンド を正しく設定していないことです。MATLAB プロンプトで mex -setup を実行し、 セットアップ中に表示されるリストから C コンパイラを選択してください。

Accelerator は、モデルの速度向上に使用するコードの生成に、Real-Time Workshop のテクノロジを使用しています。しかし、生成されたコードはモデル の速度を速める目的だけにしか使用できません。その他の目的でコードを生成し たいときは、Real-Time Workshop を使用してください。

モデル構造の変更の取り扱い

モデルのシミュレーションのために Simulink Accelerator を利用した後で、モデ ルのコンパイルバージョンである MEX- ファイルは、その後のシミュレーショ ンで利用可能です。MATLAB を終了しても、後の MATLAB または Simulink セッションで MEX- ファイルを再利用することができます。

たとえば、ブロックを追加したり削除して Simulink モデルの構造を変更する場 合、Accelerator は自動的に C コードを再生成し、既存の MEX- ファイルを更新 (上書き) します。

Accelerator の再ビルドを必要とするモデル構造の変更の例は、つぎのものです。

- 積分方法を変更する
- ブロックやブロック間の接続を追加または除去する
- 接続がベクトル化されているとしても、ブロックの入力や出力数を変更する
- モデル内の状態数を変更する
- Trigonometric Function ブロック内の関数を変更する
- Sum ブロックで用いられている符号を変更する
- S-ファンクションをインライン化するために TLC ファイルを追加する

Simulink Accelerator は、シミュレーション中に認められないモデルの変更を試み たときにワーニングを表示します。ワーニングは、カレントのシミュレーション を停止しません。モデルを変更するには、シミュレーションを停止し、変更を 行った後で、再起動してください。

変更の中には、シミュレーション中に許可されるものがあります。典型的に、 Gain ブロックの値の調整のような簡単な変更は、ワーニングを生じません。疑 わしいときは、変更を行ってみてください。ワーニングが表示されない場合は、 Accelerator は変更を認めています。

シミュレーション中にブロックがワーニングを生成しても Accelerator はワーニ ングを表示しないことに注意して下さい。例えば、ゼロ割りや数値のオーバーフ ローがあります。これは、上記に記載されているワーニングとは異なります。

Accelerator モードの性能向上

通常、Simulink Accelerator は Simulink で利用できるほとんどのブロックに対して スピードが最適化されたコードを作成します。しかし、シミュレーションを調整 したり、Accelerator の挙動を知ってさらに性能を改善することができる場合があ ります。それはつぎの場合です。

- Simulation Parameters パネル Simulation Parameters Diagnostics と Advanced パネルのオプションは、Accelerator の性能に影響を与えます。性能を向上させるためには、
 - Diagnostics パネルの Consistency checking と Bounds checking をオフにします。
 - Advanced パネルの Signal storage reuse を設定します。
- Stateflow Accelerator は Stateflow と完全に互換性がありますが、Stateflow の部 分の性能は向上されません。Stateflow ブロックがあるモデルの性能を向上さ せるには、Stateflow デバッギングとアニメーションを無効にしてください。
- ユーザ定義 S-ファンクション Accelerator は、Target Language Compiler を使っ てインライン化しなければ、S-ファンクションのシミュレーション速度を改 良できません。インライン化は、Real-Time Workshop を管理する TLC ファイ ルの作成過程に関係し、Simulink アプリケーションプログラムインタフェース (API)の不必要な呼び出しを取り除いた S-Function として C コード生成しま す。

S-ファンクションのインライン化に関する詳細は、*Target Language Compiler Reference Guide* を参照してください。これは、Math Works Web サイト www.mathworks.comやMATLABのドキュメントCDから参照することができま す。

- Simulinkおよびblocksetsが提供するS-ファンクション—Acceleratorは、Simulink およびBlocksetが提供するすべてのプロックについて互換性がありますが、 TLCファイルのインライン化に関連しないM-ファイルまたはC-MEX S-Functionプロックについてシミュレーション速度を改良しません。
- 大量データのロギング Workspace I/O、To Workspace、To File、あるいは Scope ブロックを使用している場合、大量のデータが Accelerator を減速させる

可能性があります。デシメーションするか最後の N データ点を出力するよう に制限するようにしてください。

 大規模モデル — Accelerator モード、Normal モードとも、大規模モデルを初期 化するには時間がかかります。そのため、実行時間(シングルシミュレーショ ンの開始から終了までの時間)が短いとき、Acceleratorの速度向上は最小限で す。

速度の改良を示さないブロック

Simulink Accelerator は、すべての MathWorks のブロックセットとの互換性があ りますが、Fixed Point Blockset と DSP Blockset の2つだけが Accelerator を使っ て、顕著な性能の向上を示します。

Simulink、Fixed Point Blockset および DSP Blockset ブロックを用いたモデルのシ ミュレーション性能を大きく向上できますが、現在は Accelerator によって高速 化されない Simulink と DSP Blockset ブロックのサブセットがあります。つぎの 表は、それらのブロックを示しています。

Simulink ブロック	DSP Blockset ブロック
Display	Biquadratic Filter
From File	Convolution
From Workspace	Direct-Form II Transpose Filter
Inport (ルートレベルのみ)	Dyadic Analysis Filter Bank/ Dyadic Synthesis Filter Bank
MATLAB Fcn	FIR Decimation/FIR Interpolation/ FIR Rate Conversion
Outport (ルートレベルのみ)	From Wave Device/From Wave File
Scope	Integer Delay/Variable Integer Delay
To File	Matrix Multiply/Matrix To Workspace
To Workspace	Triggered Signal To Workspace/ Triggered Signal From Workspace

表 12-1: 性能の向上を示さないプロック

Simulink プロック	DSP Blockset プロック
Transport Delay	Time-Varying Direct-Form II Transpose Filter
Variable Transport Delay	To Wave File/To Wave Device
XY Graph	Wavelet Analysis/Wavelet Synthesis
	Zero Pad

表 12-1:	性能の向上	を示さないプロ	コック	(Continued)
---------	-------	---------	-----	-------------

さらに、Accelerator は、Target Language Compiler を使ってインライン化したり、 S-ファンクションの SS_OPTION_USE_TLC_WITH_ACCELERATOR を設定しな ければ、ユーザ定義の S-Function ブロック (M, Fortran, C) を高速化しません。詳 細は、12-11 ページの "Simulink Accelerator での S- ファンクションの実行の制御" を参照してください。

Simulink Accelerator を Simulink Debugger で利用

デバッグする必要がある大規模で複雑なモデルがある場合、Simulink Accelerator は、デバッグセッションを短縮することができます。たとえば、非常に大きい時 間ブレークポイントを設定する必要がある場合、Acceleratorを使ってブレークポ イントにすばやく到達することができます。

Accelerator モードで Simulink デバッガを起動するには、

• Simulation メニューから Accelerator を選択し、MATLAB プロンプトで sldebug modelname

とタイプします。

- デバッガプロンプトで、つぎのように時間ブレークポイントを設定します。
 tbreak 10000
 continue
- ブレークポイントに到達すると、デバッガコマンド emode (execution モード)を 使って Accelerator モードと Normal モードを切り替えます。execution が Accelerator に設定されているとき、ブロックのステッピングはできません。

Simulink デバッガに関する詳しい情報は、第11章の"Simulink デバッガ"を参照してください。

Simulink Accelerator とのプログラムインタフェース

set_param, sim, accelbuild の 3 つのコマンドを使って、MATLAB プロンプトまた は M- ファイルからモデルの実行を制御することができます。この節では、これ らのコマンドの書式と利用可能なオプションを説明します。

シミュレーションモードの制御

つぎのコマンドで、MATLAB プロンプトからシミュレーションモードを制御で きます。

set_param(gcs,'simulationmode','mode')

または

set_param(modelname,'simulationmode','mode')

gcs ("get current system")を使って、現在アクティブなモデル(すなわちアクティ ブなモデルウィンドウ)のパラメータを設定することができます。また、モデル 名を明示的に指定する場合は、modelname を設定することができます。 simulationmode は、normal または accelerator のいずれかです。

高速化されたモデルのシミュレーション

つぎのコマンドを使って高速化されたモデルをシミュレートできます。

sim(gcs); % シミュレーションが完了するまで MATLAB プロンプトを使用できないようにします。

あるいは、

set_param(gcs,'simulationcommand','start');% すぐに MATLAB プロンプトが戻ります。

再度、モデルを明示的に指定したい場合は、gcsの代わりに modelname を使います。

シミュレーションと独立に Simulink Accelerator MEX-ファイルを作成 accelbuild コマンドを使うことにより、モデルを実際にシミュレートしなくでも Simulink Accelerator MEX-ファイルを作成できます。たとえば、

accelbuild f14

accelbuild 使ってバッチモードで Accelerator MEX- ファイルを作成することにより、シミュレーションの実行の前に C コードと実行可能ファイルを作成することができます。後で Accelerator を対話的に用いるときには、高速化されたシ

ミュレーションの開始時に MEX-ファイルを作成したりコンパイルする必要はありません。

accelbuild コマンドを使って、生成した MEX- ファイル内のデバッグシンボルの 変更のようなビルドオプションを指定することができます。

accelbuild f14 OPT_OPTS=-g

性能の比較

Simulink Accelerator と Normal モードの Simulink の性能を比較したい場合は、 tic,...,toc と sim コマンドを使います。F14 の例題を実行するには、つぎのコード を使います (Normal モードであることを確認してください)。

tic,[t,x,y]=sim('f14',1000);toc

elapsed_time =

14.1080

Accelerator モードでは、結果はこうなります。

elapsed_time =

6.5880

上記の結果は、233 MHz Pentium プロセッサ搭載の Windows PC で得られたものです。

Accelerator は、MEX-ファイルを再度作成しなければならない場合、実行の開始 時にチェックするので、実行時間が非常に短いモデルに対しては、Normal モー ドシミュレーションは、速くなります。これは、実行時に小規模のオーバヘッド を伴います。

Simulink Accelerator Build プロセスのカスタマイズ

典型的に、Simulink Accelerator のビルドプロセスにはカスタマイズは必要ありま せん。しかし、Accelerator はコード生成と MEX- ファイルの作成のために Real-Time Workshop と同じメカニズムを利用するので、ビルドプロセスを制御す るために 3 つのパラメータを使うことができます。

AccelMakeCommand AccelSystemTargetFile AccelTemplateMakeFile 3 つのオプションを使って、カスタム Make コマンド、システムターゲット、テ ンプレート makefiles を指定することができます。これらのパラメータは、コー ド生成プロセスの部分を管理します。これらのオプションの利用は、Real-Time Workshop のコード生成方法の理解が必要です。Make コマンド、システムター ゲットファイル、テンプレート makefile の説明については、*Real-Time Workshop User's Guide* を参照してください。これは、MathWorks Web サイト、 www.mathworks.com と MATLAB に付属するドキュメンテーション CD で参照可 能です。

これらのパラメータの設定のための構文はつぎのようになります。

set_param(gcs, 'parameter', 'string')

または、

set_param(modelname, 'parameter', 'string')

ここで、gcs ("get current system") は、カレントのアクティブなモデルで、 'parameter' は、上記の3つのパラメータのうちの1つです。string をパラメータ に対してカスタマイズした値を定義する文字列で置き換えてください。

Simulink Accelerator での S- ファンクションの実行の制御

Target Language Compiler を使った S-ファンクションのインライン化は、 Simulink Accelerator と共に用いたときに性能を向上させます。しかし、デフォル トでは、Accelerator は S-ファンクションに対してインライン化された TLC ファ イルが存在していても無視します。

このデフォルトが選択された理由の例の1つは、I/Oボードに対するデバイスド ライバ S-Function ブロックです。S-ファンクション TLC ファイルは、典型的に I/Oボードの特定のハードウェアレジスタにアクセスするために作成されます。 Accelerator はターゲットシステム上では実行しておらず、ホストシステム上のシ ミュレーションなので、S-ファンクションに対してインライン化された TLC ファイルの利用を避けなければなりません。

その他の例は、TLC ファイルと S- ファンクションの MEX- ファイルバージョン が作業ベクトル、パラメータの利用、初期化において非互換であることです。

インライン化された S- ファンクションがこれらの問題によって複雑化されてい ない場合、S- ファンクションの mdlInitializeSizes 関数の SS_OPTION_USE_TLC_WITH_ACCELERATOR を指定することによって、S-ファンクション MEX- ファイルの代わりに TLC ファイルを使うことができます。

設定時に Accelerator は、インライン化されたファイルを利用し、完全な性能の向上が実現されます。

たとえば、

```
static void mdlInitializeSizes(SimStruct *S)
{
    /* Code deleted */
    ssSetOptions(S, SS_OPTION_USE_TLC_WITH_ACCELERATOR);
}
```

Model Differencing Tool

Model Differencing Tool は、2 つの Simulink モデルの差分を見つけ表示します。これによって、たとえば同じモデルのバージョンの違いなどを素早く決定することができます。

注意 このツールを使用するには、Simulink Performance Tools がインストールされていなければなりません。

このツールを使用するには、比較する2つのモデルのうちの1つを開き、 SimulinkのToolメニューから Model differences を選択してください。Model Differencing Toolは、Select Second Model ダイアログボックスとともに表示されます。

Select Second Model			? ×
Look jn: 🔄 win32	•		*
🚞 mbuildopts	🖻 atlas_PIII.exp	4	Common_conte
a mexopts	🛋 atlas_PIII.lib		common_conte
🔊 atlas_Athlon.dll	🖻 blas.spec	4	🔊 comp_ja.dll
atlas_Athlon.exp	🔊 clbs110.dll		🖻 comp_ja.map
🖻 atlas_Athlon.lib	💽 cmex.bat		🛋 compiler.csf
🔊 atlas_PIII.dll	🛋 common_context.csf	٩	🔊 compiler.dll
•			F
File <u>n</u> ame:			<u>O</u> pen
Files of type: All Files (*.*)		-	Cancel

Select Second Model ダイアログボックスから、比較したいもう一方のモデルを選択します。Model Differences Tool は、2番目のモデルを開き、必要に応じて、 Model Differences Tool と2つのモデルを並べて表示します。



Model Differences Tool は3つのパネルから構成されます。左上のパネルは、最初のモデルの内容を拡張可能なリストで表示します。右上のパネルは、2番目のモデルの内容を表示します。色は、2つのモデルの差分を示しています。

- ・ 青色は、2つのモデルのうち片方だけにあるブロックです。
- 赤色は、両方のモデルに存在するがパラメータ値や(サブシステムの場合)コン テンツの値が異なるブロックです。
- 緑色は、両方のモデルで同じブロックです。

それぞれのパネルのブロックをクリックすると、モデル内の対応するブロックの アイコンがハイライトされます。下のパネルは、両方のモデルに存在するブロッ クを選択したときに、ブロックのバージョン間のパラメータの相違点を表示しま す。

表示オプション

このツールには、いくつかの表示オプションがあります。Options メニューから Show items with differences only を選択すると、2 つのモデルで差分のないブロック が省かれて表示されます。Include only non-graphical differences を選択すると、パ ラメータや内容に違いがあるブロックだけが表示されます。ブロックの位置や背 景色のようなグラフィックの違いだけがあるサブシステムブロックは、このオプ ションによって除かれます。

モデルの差分をレポート

View メニューから HTML Report を選択すると、2 つもモデルの差異をまとめた HTML 形式のレポートが表示されます。

Simulir	nk/Stateflow	Model D	ifference	es Repo	rt - Micro	soft Inter	rnet Exp	olorer pro	ovided by	The Mat	hW	. 🗆 ×
<u> </u>	dit <u>V</u> iew <u>C</u>	ào F <u>a</u> vo	rites <u>H</u> e	lp								ê
÷	⇒		\$	â	Q		ک	_ Q			A	
Back	Forward	Stop	Refresh	Home	Search	Favorites	History	Channels	Fullscreen	Mail	Fonts	Prin
Linko 🎒	EmposiDE	262659 3EAO Page			DE Mol Av	مىشە 🐴 ס	loooword C	hango 🗿			A TTM	
j Links 🛃		Ji Aqirage	- Insibi	aye 🛃		anse 🔤 i	asswoid C	nange 🛃	rreair iayei	- Ineserve	• • 1116	smat r
Sim	ulink/S	Statef	low	Mod	lel D	oiffer	ence	es Re	port			-
Report d	ate: 11-Sep-2	2000										
Model 1: Model 2:	vdp vdp1											
Number	of objects co	ntaining di	fferences:	3								
Objects	Containing	Differe	nces									
Object Type	Model 1			N	fodel 2							
Block Diagram	vdp			<u>v</u>	<u>dp1</u>							
Block	vdp/Mu			V	dp1/Mu							
Block	n/a			V	dp 1/Dispi	lay						
Differer Parame	nces betwee ter	en vdp a	nd vdp1	L	vdp	1]			
Name		vdp			vdp	1						_
Created		Fri Aug 1	8 16:03:	19 2000	Mot	n May 22	10:30:3	2 2000				
Creator		mani			paul	k						
LastMo	difiedBy	mani			paul	k						
LastMoo	difiedDate	Fri Aug 1	8 16:05:4	19 2000	Mor	n Sep 11	15:20:50	2000				
ModelV	ersionFormat	1.%			1.%							
Lines		2			2							
2]										My Compute	a a	•

このレポートでは最初に、モデル間のすべての差分ブロックがリスト表示されます。続いて、モデル間で差分のあるブロック毎のレポートが続きます。

プロファイラ

Simulinkのシミュレーションプロファイラは、モデル解析中のパフォーマンス データを収集したり、そのデータに基づいたシミュレーションプロファイルとい うレポートを作成します。プロファイラによって作成されるシミュレーションプ ロファイルには、Simulinkがモデルをシミュレーションするのに必要な関数ごと に実行に要した時間が表示されます。プロファイルによってモデルのどのパーツ がシミュレーションを実行するにあたって最も時間を必要としたかが分かり、こ の結果より、モデルのどの部分に焦点をあてて、最適化する努力を行えば良いか を決めることができます。

注意 プロファイラを使用するには、Simulink Performance Tools をインストール する必要があります。

どのようにしてプロファイラが働くか

つぎの疑似コードは、プロファイラが基本としている実行モデルの要約です。

Sim()

ModelInitialize(). ModelExecute() for t = tStart to tEnd Output() Update() Integrate() Compute states from derivs by repeatedly calling: MinorOutput() MinorDeriv() Locate any zero crossings by repeatedly calling: MinorOutput() MinorZeroCrossings() EndIntegrate Set time t = tNew. EndModelExecute ModelTerminate EndSim

この権	既念モデルに基づき、	Simulink は、こ	⊃ぎの表の関数	タをモデルや関数に	より、
0回、	1回あるいはそれ以	上の回数呼び出	し、Simulink	モデルを実行します	Γ.

関数	目的	レベル
sim	モデルをシミュレーションします。 この最上位レベルの関数はモデルを シミュレーションするために必要な 関数を呼び出します。この関数に要 する時間は、モデルをシミュレー ションする総時間です。	システム
ModelInitialize	シミュレーションのためにモデルの 準備	システム
ModelExecute	シミュレーション開始から終了まで の時間ステップ毎に、output、 update、integrate などの関数がそれぞ れのブロックのために呼び出されて、 モデルが実行されます。	システム
Output	カレント時間ステップでの、1 ブ ロックの出力を計算します。	ブロック
Update	カレント時間ステップでのブロック の状態を更新します。	ブロック
Integrate	カレント時間ステップで、状態導関 数を積分することによってブロック の連続状態を計算します。	ブロック
MinorOutput	マイナー時間ステップでブロック出 力を計算します。	ブロック
MinorDeriv	マイナー時間ステップで、ブロック の状態導関数を計算します。	ブロック
MinorZeroCrossings	マイナー時間ステップで、ブロック のゼロクロッシング値を計算します。	ブロック

関数	目的	レベル
ModelTerminate	メモリを解放し、すべてのシミュ レーションの最後を一掃します。	システム
Nonvirtual Subsystem	それぞれのブロックごとに output, update, integrate などの関数を呼び出 すことによって、カレント時間ス テップで、非バーチャルサブシステ ム(3-13ページの"Atomic サブシステ ムと仮想サブシステム"を参照してく ださい)の出力を計算します。この 関数に要した時間は、非バーチャル サブシステムを実行するのに必要な 時間です。	ブロック

プロファイラは、それぞれの関数の呼び出しを実行するのに必要な時間を算出 し、各関数にどれくらいの時間がかかったかの詳細なレポートをモデルの最後に 作成します。

プロファイラを利用可能にする

モデルをプロファイルするためには、モデルを開き、Simulinkの Tools メニュー から Profiler を選択します。その後、シミュレーションを開始します。シミュ レーションが終了するとき、Simulink は MATLAB help browser 上に、モデルのシ ミュレーションプロファイルを作成し表示します。

シミュレーションプロファイル

Simulink は MATLAB ワーキングディレクトリにシミュレーションプロファイル を保存します。



レポートは、概要部分と詳細なレポート部分からなります。

概要部分

概要ファイルには、つぎの性能の総計が表示されます。

項目	説明
Total Recorded Time	モデルを実行するのに要した総時間
Number of Block Methods	ブロックレベル関数 (例えば、Output()) を実 施した総回数

項目	説明
Number of Internal Methods	システムレベル関数 (例えば、ModelExecute) を実施した総回数
Number of Nonvirtual Subsystem Methods	非バーチャルサブシステム関数を実施した総 回数
Clock Precision	プロファイラの時間測定の精度

概要部分には、シミュレーションで呼び出される各関数の概要プロファイルが示 されます。それぞれの関数をリスト表示するために、概要プロファイルは、次の 情報を記します。

項目	詳細
Name	関数の名前。この項目はハイパーリンクです。クリックする と、この関数の詳細なプロファイルが表示されます。
Time	関数のすべての呼び出しを実行するのに要した総時間を絶対 値や総解析時間のパーセンテージで表示。
Calls	関数が呼び出された回数
Time/Call	関数の呼び出しに要した平均時間、この関数によって呼び出 された関数に要した時間を含みます。
Self Time	この関数を実行するのに要した平均時間、この関数によって 呼び出された関数に要した時間は省きます。
Location	関数が呼び出され実行されたブロックやモデルを示します。 この項目はハイパーリンクされています。クリックすると、 モデルダイアグラムの対応するアイコンがハイライトされま す。このハイパーリンクは、MATLAB help browser でプロ ファイルを見ているときにのみ使用できます。

詳細なプロファイル部分

この部分には、モデルをシミュレーションするために呼び出されるそれぞれの関数ごとの詳細なプロファイが含まれます。それぞれの詳細なプロファイルには、 関数の概要プロファイルで示されたすべての情報が含まれています。また、詳細 なプロファイルには、プロファイルされた関数を呼び出した関数(親関数)やプ ロファイルによって呼び出された関数(子関数)が表示されます。親関数や子関 数の名前をクリックすると、その関数の詳細なプロファイルが表示されます。

Model Coverage Tool(モデル補償範囲ツール)

Model Coverage Tool は、あるモデルにおいて、あるテストケースが影響を及ぼし たシミュレーション経路の範囲を検出します。テストケースが影響を及ぼした経 路のパーセンテージを model coverage(モデル補償範囲)と呼びます。モデル補償 範囲は、あるテストがどれくらい十分にモデルをテストしたかを算出します。そ のため、Model Coverage Tool によって、モデルテストの有効性を調べることがで きます。

注意 Model Coverage Tool を使用するためには、Simulink Performance Tools をイ ンストールする必要があります。

どのようにして Model Coverage Tool が働くか

Model Coverage Tool は、モデル内で切り替え点の役割をするブロックの実行を分析します。切換え点を表わすブロックタイプには、つぎのブロックがあります。

- Switch
- Multiport Switch
- Triggered subsystem (Trigger ブロックを含むサブシステム)
- Enabled subsystem (Enable ブロックを含むサブシステム)
- Absolute Value
- Saturation

モデルが Stateflow のチャートを含んでいる場合は、チャートの状態や遷移も分析します。ツールは、シミュレーションが実行されている間中、分岐ブロックの 状態や遷移の変化を記録します。シミュレーションの最後に、それぞれの切換え 点をあらわすブロックや状態や遷移ごとに、実際の分岐に対する潜在的な分岐率 を計算します。

Model Coverage Tool の使用

Model Coverage Tool で効果的なテストを行うために

1 モデルに対し、1つ以上のテストケースを行います(12-24ページの"テスト ケースの作成と実行"を参照してください)。

- 2 モデルの挙動が正しいか検証するためのテストケースを実行します。
- 3 Simulink が出力するカバレージレポートを分析して検討します。
- 4 カバレージレポートの情報を使って、補償範囲が増えるようにテストケース を修正したり、現在のテストケース集合ではカバーされない領域をカバーす るような新しいテストケースを加えます。
- 5 テストケース集合が補償範囲を満足するまで、この過程を繰り返します。

注意 Simulink には、モデルテストの検査に Model Coverage Tool を使用したオン ラインデモがあります。このデモを実行するには、MATLAB コマンドプロンプ トに simcovdemo と入力してください。

テストケースの作成と実行

Test Coverage Tool には、テストケースを作成し実行するために、2 つの MATLAB コマンド cvtest と cvsim が用意されています (12-33 ページの "cvsim" と 12-33 ページの "cvtest" を参照してください)。

Coverage Tool を対話形式で実行することも可能です。そのためには、Simulink の Tools メニューから Coverage Settings を選択します。Simulink は、Coverage Settings ダイアログボックス (12-29ページの"カバレージの設定ダイアログボック ス"を参照してください)を表示します。Enable Coverage Reporting をチェック し、ダイアログを消すために OK を選択してください。そして、Simulation メ ニューから Start を選択するか、Simulink ツールバーのスタートボタンを選択し てください。

デフォルトでは、Simulink は covdata という名前のワークスペースオブジェクト としてデータを保存し、シミュレーションの最後に HTML レポートとしてデー タを表示します。その他、補償範囲データを作成、保存、レポート作成するオプ ションを選択することができます。 注意 モデルカバレージレポート作成とアクセラレーションオプションを利用可 能な状態でシミュレーションを実行することはできません。アクセラレーション が利用可能な場合、Simulink はカバレージレポートを利用不可にします。モデル が Stateflow ライブラリチャートへのリンクを含み、Model Converage Tool のカバ レージレポートにチャートも含めたい場合、シミュレーションを開始する前に、 ライブラリチャートを開いておく必要があります。もし、リファレンスライブラ リチャートが開かれてない場合、ツールはチャートを省略します。

カバレージレポート

Model Converage Tool で作成されるカバレージレポートはつぎの項目からなります。

カバレージサマリ

カバレージサマリ節は、3つのサブ節で構成されます。



- "Summary"の節には、モデル全体に対するすべてのテストケースの総有効範囲が記述されます。
- "Tests" の節には、各テストケースにおけるシミュレーションの開始時間と終 了時間、およびシミュレーション直前のセットアップコマンドがリストされ

ます。それぞれのテストケースを示すヘッダーには、たとえば "Test throttle," のように、cvtest コマンドで与えたテストケースのラベルが含まれます。

 "Model Systems"の節には、各サブシステムの結果の要約が記述されます。サ ブシステムの名前をクリックすると、サブシステムの詳細レポートが表示さ れます。

Details (詳細)節

"Details" 節では、モデルの補償範囲結果の詳細がレポートされます。



"Details" 節には、モデル全体の結果の要約に続いて、モデルに含まれるサブシス テムやチャートの要約が記述されています。それぞれのサブシステムやチャート は、サブ節に続きます。モデル要約内のサブシステムやチャートの名前をクリッ クすると、そのサブシステムやチャートの詳細レポートが表示されます。

サブシステムのレポート

それぞれのサブシステムの節は、サブシステムのテスト補償範囲の結果の要約 と、サブシステムのリストからなります。概要に続き、サブシステム内の判定点 になっているそれぞれのブロックのブロックレポートが続きます。

ブロックのレポート

それぞれのブロックの節は、信号通過が可能な判定経路と、実際のテストシミュ レーションがそこを通った回数をリストしている表からなります。信号が通過し なかった経路をもつブロックは、レポート中で赤くハイライトされます。ブロッ ク名をクリックすると、Simulink は、ブロックを含んでいるブロックダイアグラ ムを表示します。同時に、ブロックを見つけ易くするために、ダイアグラムの中 のブロックもハイライトします。

注意 モデルへのハイパーリンクは、カレント MATLAB セッションでのみ有効 です。その後のセッションでも、ハイパーリンクを保存するためには、レポート を生成しなければなりません。

それぞれのブロックのセッションでは、backward(後ろへ)と forward(前へ)の 矢印が表示されます。forward 矢印をクリックすると、カバーされなかった経路 をリストしているレポートの次の節が表示されます。back 矢印をクリックする と、前のカバーされなかった経路が表示されます。

チャートのレポート

Stateflow チャートの詳細レポートも同じような書式で、チャート内のそれぞれの状態と遷移の決定表からなります。

カバレージの設定ダイアログボックス

Coverage Settings ダイアログボックスで、モデルのカバレージレポートオプションを選択することができます。

🥠 Coverage Settings - vdp 📃 🗖 🗙
Enable Coverage Reporting
Coverage Instrumentation Path (// for whole model)
1
Browse
Save Results
Save to workspace (cvdata object): covdata
Increment variable name with each simulation (var1, var2,)
Reporting
Generate HTML report
Additional data to include in report (cvdata objects):
OK Cancel Help Apply

ダイアログボックスでは、つぎのオプションが設定できます。

Enable converage Reporting(カバレージレポートの利用)

Simulinkh は、シミュレーション中のモデルカバレージデータを集めレポートします。

Coverage Instrumentation Path(カバレージ構成パス)

Simulink がカバレージデータを集めてレポートをするためのサブシステムのパス のことです。デフォルトでは、Simulink は全体モデルのカバレージデータを生成 します。カバレージレポートを特定のサブシステムに制限する場合は、Browse を選択してください。

Simulink は、System Selector ダイアログを表示します。

📣 System Selector 📃 🗖 🗙	
 fuelsys EGO sensor MAP sensor engine speed fuel rate controller speed sensor throttle command throttle sensor 	
OK Cancel	

カバレージレポートを利用できるようにしたいサブシステムを選択してください。ダイアログを消すには、OKをクリックしてください。

Save to workspace(ワークスペースに保存)

Simulink によって生成されたカバレージデータが含まれているワークスペースオ ブジェクトの名前

Increment variable name with each simulation(各シミュレーションごとの 変数名の増分)

このオプションを選択すると、カバレージデータオブジェクトの名前をシミュ レーションごとに増やします。これによって、カレントシミュレーションが、前 の実行結果を上書きしません。

Generate HTML report(HTML レポートの生成)

カバレージデータを含む HTML 形式のレポートを作成し、シミュレーションの 最後に MATLAB ヘルプブラウザにレポートを表示します。

Additional data to include in report(追加データをレポートに含める) カレントカバレージデータとともにカレントレポートに追加する以前の解析のカ バレージデータの名前。このオプションと前のオプションによって、複数のシ ミュレーション実行結果が含まれているレポートが作成されます。

Model Coverage コマンド

cvhtml

cvdata オブジェクトの HTML レポートを作成します。

cvhtml(file,data)

cvdata オブジェクトの data からカバレージ結果の HTML レポートを作成します。 レポートは file に書き込まれます。

cvhtml(file,data1,data2,...)

複数のデータオブジェクトが結合したレポートを作成します。それぞれのオブ ジェクトの結果は、1 列ごとに表示されます。それぞれのデータオブジェクト は、同じルートサブシステムに対応していなければならず、対応していない場合 は関数がエラーを出力します。

cvhtml(file,data,data2,...,detail)

0 から 3 の整数で、detail のレポート詳細レベルを指定します。より大きな数字 でより詳細になります。デフォルトは 2 です。

cvload

カバレージテストのロードとファイル作成

[TESTS, DATA] = CVLOAD(FILENAME)

FILENAME.CVT テキストファイルに保存されているテストとデータをロードし ます。テストが正常にロードされると、cvtest オブジェクトのセル配列 TESTS に戻されます。DATA は、正常にロードされた cvdata オブジェクトのセル配列 です。DATA は TESTS と同じサイズですが、特定のテストが結果を持たないと きは空要素を含みます。

特に配慮しなければならないこと。

- カバレージデータベースに同じ名前のモデルが存在する場合、ファイルから 矛盾のない結果だけがロードされ、重複が起こらないように既存のモデルを 参照します。
- ファイルから参照される Simulink モデルは開いているけれども、カバレージ データベースに存在しない場合、カバレージツールは既存モデルヘリンクし ます。

同じモデルを参照する数個のファイルをロードする場合、より最近のファイルに合っている結果だけがロードされます。

cvreport

オブジェクトの情報をレポートします。このコマンドは、つぎのような書式です。

cvreport(file,data)

cvdata オブジェクトデータに補償範囲結果のテキストレポートを作成します。レ ポートは、ファイルに書き込まれます。ファイルが空の場合、レポートはコマン ドプロンプトに表示されます。

cvreport(file,data1,data2,...)

いくつかのテストオブジェクトの結果を1つにまとめて作成します。それぞれの オブジェクトの結果は、1列ごとに表示されます。それぞれのデータオブジェク トは、同じルートサブシステムに対応していなければならず、そうでない場合 は、エラーが生成されます。

cvreport(file,data1,data2,...,detail)

0 から 3 の整数で、detail 値のレポートの詳細レベルを指定します。より大きな 数字でより詳細になります。デフォルトは 2 です。

cvsave

カバレージテストと結果をファイルに保存します。

cvsave(filename,model)

filename.cvt テキストファイルに model に関係したすべてのテストや結果を保存します。

cvsave(filename, test1, test2, ...)

filename.cvt テキストファイルに指定したテストを保存します。参照モデルの情報 も保存されます。

cvsave(filename, data1, data2, ...)

指定したデータオブジェクト、データオブジェクトを作成したテストおよび参照 モデルの構造をテキストファイル filename.cvt に保存します。 cvsim

テストケースの実行

注意 このコマンドを使用するために、モデルカバレージレポート(12-24 ページの"テストケースの作成と実行"参照)を利用可能にする必要はありません。

このコマンドはつぎの書式を取ることができます。

data = cvsim(test)

対応するモデルのシミュレーションを実行するために、cvtest オブジェクトの test が実行されます。結果は cvdata オブジェクトに戻されます。

[data,t,x,y] = cvsim(test)

シミュレーション時間ベクトルt、状態値x、そして出力値yを戻します。

[data,t,x,y] = cvsim(test, timespan, options)

デフォルトシミュレーション値を上書きします。詳細は、sim コマンドを参照し てください。

[data1, data2, ...] = cvsim(test1, test2, ...)

テストの集合を実行し、cvdata オブジェクトに結果を戻します。

[data1,t,x,y] = cvsim(root, label, setupcmd)

cvtest オブジェクト作成、実行します。

cvtest

テストケースを作成します。このコマンドの書式はつぎのとおりです。

test = cvtest(root, label, setupcmd)

ここで、root はテストされるモデルやサブシステムの名前またはハンドルです。 label はテストケースを見分けるための文字列です。setupcmd は、モデルを実行 する前に、ベースワークスペースで cvsim が実行する MATLAB コマンドです。 2 番目以降の 2 つの引数はオプションです。cvtest コマンドは、テストケースに 対応したハンドルを戻します。
モデルとブロックパラメー タ

はじめに	 	 . A-2
モデルパラメータ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	 	 . A-3
共通ブロックのパラメータ..............	 	 . A-7
プロック固有のパラメータ................	 	 A-10
マスクパラメータ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	 	 A-26

はじめに

この付録では、モデル、ブロック、マスクのパラメータをリストします。パラ メータをリストしている表には、set_param コマンドを使ってコマンドラインか らモデルの修正が可能な情報が含まれています。このコマンドの詳細は、10-27 ページの set_param を参照してください。

モデルパラメータ

つぎの表は、モデルを記述するパラメータの一覧と説明です。パラメータは、付 録 B. で説明するモデルファイルで定義されている順番に表示されます。表には、 4-71 ページの"コールバックルーチンの使用法"で記述されているモデルのコー ルバックパラメータも含まれています。説明の欄には、Simulation Parameters ダイアログボックスのどこでその値を設定することができるかを示しています。 シミュレーションパラメータであるモデルパラメータの詳細については、5-8 ページの"Simulation Parameters ダイアログボックス"に記述してあります。表の 後に、パラメータの変更方法についての例を示します。

パラメータの値はコーテーションで囲んだ文字列として指定しなければなりません。文字列の内容はパラメータに依存し、数値(スカラ、ベクトル、行列)、変数名、ファイル名、または特定の値を与えることができます。値の欄には、必要な値のタイプ、(垂直バーで区切られた)設定可能な値、および中括弧で囲んだデフォルト値が示してあります。

表A-1: モデルパラメーク	表 A-1:	モデ	ルパラ	メー	タ
----------------	--------	----	-----	----	---

パラメータ	説明	値
AbsTol	絶対誤差許容値	スカラ {1e-6}
AlgebraicLoopMsg	代数ループの診断	none {warning} error
ArrayBoundsChecking	配列範囲チェックの利用	'none' 'warning' 'error'
BooleanDataType	Boolean モードの使用の可能性	on {off}
BufferReuse	ブロック I/O バッファの再使用の可能性	{on} off
CloseFcn	閉じる際のコールバック	コマンドまたは変数
ConfigurationManager	モデルのコンフィグレーションマネージャ	文字列
ConsistencyChecking	整合性のチェック	on {off}
Created	モデルの作成日付と時刻	文字列
Creator	モデル作成者名	文字列
Decimation	間引きファクタ	スカラ {1}
Description	モデルの説明	文字列
ExternalInput	時間および入力変数名	スカラまたはベクトル [t, u]

表 A-1: モデルパラメータ (Continued)

パラメータ	説明	値
FinalStateName	最終状態名	変数 {xFinal}
FixedStep	固定ステップサイズ	スカラ {auto}
InitialState	初期状態名または値	変数またはベクトル {xInitial}
InitialStep	初期ステップサイズ	スカラ {auto}
InvariantConstants	不変定数の設定	on {off}
LimitDataPoints	出力の制限	on {off}
LoadExternalInput	ワークスペースから入力のロード	on {off}
LoadInitialState	初期状態のロード	on {off}
MaxDataPoints	保存する出力データの最大数	スカラ {1000}
MaxOrder	ode15s に対する最大次数	1 2 3 4 {5}
MaxStep	最大ステップサイズ	スカラ {auto}
MinStepSizeMsg	最小ステップサイズの診断	{warning} error
ModelVersionFormat	モデルのバージョン番号の書式	文字列
ModifiedBy	モデルの最終修正者	文字列
ModifiedDateFormat	修正日付の書式	文字列
Name	モデル名	文字列
ObjectParameters	モデルパラメータの名前 / 属性	構造体
OutputOption	出力オプション	追加出力の生成 { 出力リファイン } 指定出力のみの生成
OutputSaveName	シミュレーション出力名	変数 {yout}
OutputTimes	選択された OutputOption に対する値	ベクトル {[]}
PaperOrientation	印刷用紙の方向	portrait {landscape}
PaperPosition	紙面上でのダイアグラムの位置	[left, bottom, width, height]

パラメータ	説明	値
PaperPositionMode	用紙の位置のモード	auto {manual}
PaperSize	PaperUnits での PaperType のサイズ	[width height] (read only)
РарегТуре	印刷用紙のタイプ	{usletter} uslegal a0 a1 a2 a3 a4 a5 b0 b1 b2 b3 b4 b5 arch-A arch-B arch-C arch-D arch-E A B C D E tabloid
PaperUnits	印刷用紙のサイズの単位	normalized {inches} centimeters points
PostLoadFcn	ポストロードする際のコールバック	コマンドまたは変数
PreLoadFcn	プリロードする際のコールバック	コマンドまたは変数
Refine	リファインファクタ	スカラ {1}
RelTol	相対誤差許容値	スカラ {1e-3}
SampleTimeColors	Sample Time Colors メニューオプション	on {off}
SaveFcn	保存する際のコールバック	コマンドまたは変数
SaveFinalState	最終状態の保存	on {off}
SaveFormat	MATLAB ワークスペースにデータを保存 する際に使用される書式	配列 構造体 時間付き構造体
SaveOutput	シミュレーション出力の保存	{on} off
SaveState	状態の保存	on {off}
SaveTime	シミュレーション時間の保存	{on} off
ShowLineWidths	Show Line Widths メニューオプション	on {off}
SimParamPage	表示する Simulation Parameters ダイアログ ボックスのページ(最後に表示されたペー ジ)	{Solver} ワークスペース I/O 診断
Solver	ソルバ	{ode45} ode23 ode113 ode15s ode23s ode5 ode4 ode3 ode2 ode1 固定ス テップ離散 可変ステップ離散
StartFcn	シミュレーション開始時のコールバック	コマンドまたは変数

表 A-1: モデルパラメータ (Continued)

パラメータ	説明	值
StartTime	シミュレーションの開始時間	スカラ {0.0}
StateSaveName	状態出力名	变数 {xout}
StopFcn	シミュレーション終了時のコールバック	コマンドまたは変数
StopTime	シミュレーションの終了時間	スカラ {10.0}
TimeSaveName	シミュレーション時間名	变数 {tout}
UnconnectedInputMsg	未接続の入力端子の診断	none {warning} error
UnconnectedLineMsg	未接続のラインの診断	none {warning} error
UnconnectedOutputMsg	未接続の出力端子の診断	none {warning} error
Version	モデルを修正するために使われた Simulink のバージョン(参照のみ)	(release)
WideVectorLines	Wide Vector Lines メニューオプション	on {off}
ZeroCross	厳密なゼロクロッシング検出 (3–14 ページ の " ゼロクロッシングの検出 " を参照)	{on} off

表 A-1: モデルパラメータ (Continued)

以下の例では、mymodel システムに対するモデルパラメータの設定方法を示します。

つぎのコマンドは、シミュレーションの開始時間と終了時間を設定します。

set_param('mymodel','StartTime','5','StopTime','100')

つぎのコマンドは、ソルバを ode15s に設定し、最大次数を変更します。

set_param('mymodel','Solver','ode15s','MaxOrder','3')

つぎのコマンドは、SaveFcn コールバックに関連付けます。

set_param('mymodel','SaveFcn','my_save_cb')

共通ブロックのパラメータ

つぎの表は、4-71 ページの"コールバックルーチンの使用法"に記述されている コールバックパラメータを含む、すべての Simulink ブロックに共通のパラメー タをリストしています。表の後に、これらのパラメータを変更するコマンドの例 を示します。

表 A-2: 共通のブロックパラメータ

パラメータ	説明	值
AttributesFormat String	ブロック線図内のブロックの下 に表示される設定したパラメー タ	文字列
BackgroundColor	ブロックアイコンの背景色	black {white} red green blue cyan magenta yellow gray lightBlue orange darkGreen
BlockDescription	ブロックの記述	テキスト
BlockType	ブロックタイプ	テキスト
CloseFcn	閉じる際のコールバック	MATLAB 表現
CompiledPortWidths	端子幅の構造体	スカラまたはベクトル
CopyFcn	コピーする際のコールバック	MATLAB 表現
DeleteFcn	削除する際のコールバック	MATLAB 表現
Description	ユーザ指定の記述	テキスト
DialogParameters	ブロックパラメータダイアログ 内のパラメータの名前 / 属性	構造体
DropShadow	ドロップシャドウの表示	{off} on
FontAngle	フォントの角度	(システムに依存) {normal} italic oblique
FontName	フォント	{Helvetica}
FontSize	フォントサイズ	{10}
FontWeight	フォントの重み	(システムに依存) light {normal} demi bold
ForegroundColor	ブロック名、アイコン、アウト ライン、出力信号、信号ラベル	{black} white red green blue cyan magenta yellow gray lightBlue orange darkGreen

パラメータ	説明	值
InitFcn	初期化する際のコールバック	MATLAB 表現
InputPorts	入力端子位置の配列	[h1,v1; h2,v2;]
LinkStatus	ブロックのリンク状態	none resolved unresolved implicit
LoadFcn	ロードする際のコールバック	MATLAB 表現
ModelCloseFcn	モデルを閉じる際のコールバッ ク	MATLAB 表現
Name	ブロック名	文字列
NameChangeFcn	ブロック名を変更する際のコー ルバック	MATLAB 表現
NamePlacement	ブロック名の位置	{normal} alternate
ObjectParameters	ブロックのパラメータの名前/ 属性	構造体
OpenFcn	開く際のコールバック	MATLAB 表現
Orientation	ブロックの向き	{right} left down up
OutputPorts	出力端子位置の配列	[h1,v1; h2,v2;]
Parent	ブロックを所有するシステムの 名	string
ParentCloseFcn	親サブシステムを閉じる際の コールバック	MATLAB 表現
Position	モデルウィンドウ内のブロック の位置	ベクトル [left top right bottom] コーテーションでは <i>囲みません。</i>
PostSaveFcn	ポストセーブする際のコール バック	MATLAB 表現
PreSaveFcn	プリセーブする際のコールバッ ク	MATLAB 表現
Selected	ブロックの選択状態	on {off}
ShowName	ブロック名の表示	{on} off

表 A-2: 共通のプロックパラメータ (Continued)

パラメータ	説明	值
StartFcn	シミュレーションを開始する際 のコールバック	MATLAB 表現
StopFcn	シミュレーションを停止する際 のコールバック	MATLAB 表現
Tag	ユーザ定義文字列	,,
Туре	Simulinkオブジェクトタイプ(参 照のみ)	'block'
UndoDeleteFcn	ブロックの削除をアンドゥする 際のコールバック	MATLAB 表現
UserData	任意の MATLAB データタイプ (mdl ファイルには保存されま せん)	[]

表 A-2: 共通のブロックパラメータ (Continued)

以下の例では、これらのパラメータを変更する方法を示します。

つぎのコマンドは、mymodel システム内の Gain ブロックの向きを変更して、反対方向(右から左)を向くようにします。

set_param('mymodel/Gain','Orientation','left')

つぎのコマンドは、OpenFcn コールバックを mymodel システムの Gain ブロック に関連付けます。

set_param('mymodel/Gain','OpenFcn','my_open_cb')

つぎのコマンドは、mymodel システムの Gain ブロックの Position パラメータを 設定します。ブロックは、幅が 75 ピクセルで、高さが 25 ピクセルです。位置ベ クトルは、絶対にコーテーションで*囲みません*。

set_param('mymodel/Gain','Position',[50 250 125 275])

ブロック固有のパラメータ

つぎの表には、すべての Simulink ブロックに対するブロック固有のパラメータ をリストしています。set_param コマンドでプロックパラメータを設定する際に は、BlockType パラメータを指定することによってブロックを識別します。 BlockType は、プロック名の後の括弧の中に表示されます。

注意 表にはマスクされたブロックに対する MaskType パラメータ値が含まれて いますが、詳細情報は組み込みブロックに対するもののみで、マスクされたブ ロックに対するものは含まれていません。詳細については、A-26 ページの"マス クパラメータ"を参照してください。

ダイアログボックスプロンプトの欄には、ブロックのダイアログボックス上のパ ラメータに対するプロンプトのテキストが示してあります。値の欄には、必要な 値のタイプ(スカラ、ベクトル、変数)、(垂直バーで区切った)設定可能な値、 および(中括弧で囲んだ)デフォルト値が示してあります。

表 A-3: Sources ライブラリブロックパラメータ

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
Band-Limited White Noise (Continuous W	/hite Noise) (マスクされた)	
Chirp Signal (chirp) (masked)		
VectorParams1D	Interpret vector parameters as 1-D	off {on}
Clock (Clock) (no block-specific parameters)		
Constant (Constant)		
Value	Constant value	スカラまたはベクトル {1}
VectorParams1D	Interpret vector parameters as 1-D	off {on}
Digital Clock (DigitalClock)		

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
SampleTime	Sample time	スカラ (sample period) {1} またはベクトル [period offset]
Digital Pulse Generator		
VectorParams1D	Interpret vector parameters as 1-D	off {on}
From File (FromFile)		
FileName	Filename	ファイル名 {untitled.mat}
From Workspace (FromWorkspace)		
VariableName	Matrix table	行列 {[T,U]}
Pulse Generator (Pulse Generator) (masked)		
VectorParams1D	Interpret vector parameters as 1-D	off {on}
Ramp (Ramp) (masked)		
VectorParams1D	Interpret vector parameters as 1-D	off {on}
Random Number (RandomNumber)		
Seed	Initial seed	スカラまたはベクトル {0}
VectorParams1D	Interpret vector parameters as 1-D	off {on}
Repeating Sequence (Repeating table) (ma	usked)	
Signal Generator (SignalGenerator)		
WaveForm	Wave form	{sine} square sawtooth random
Amplitude	Amplitude	スカラまたはベクトル {1}
Frequency	Frequency	スカラまたはベクトル {1}
Units	Units	{Hertz} rad/sec

表 A-3: Sources ライブラリプロックパラメータ (Continued)

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
VectorParams1D	Interpret vector parameters as 1-D	off {on}
Sine Wave (Sin)		
Amplitude	Amplitude	スカラまたはベクトル {1}
Frequency	Frequency	スカラまたはベクトル {1}
Phase	Phase	スカラまたはベクトル {0}
SampleTime	Sample time	スカラ (サンプル周期) {-1} またはベクトル [period offset]
VectorParams1D	Interpret vector parameters as 1-D	off {on}
Step (Step)		
Time	Step time	スカラまたはベクトル {1}
Before	Initial value	スカラまたはベクトル {0}
After	Final value	スカラまたはベクトル {1}
VectorParams1D	Interpret vector parameters as 1-D	off {on}
Uniform Random Number (Uniform Rand	omNumber)	
Minimum	Minimum	スカラまたはベクトル {-1}
Maximum	Maximum	スカラまたはベクトル {1}
Seed	Initial Seed	スカラまたはベクトル {0}
SampleTime	Sample Time	スカラまたはベクトル {0}
VectorParams1D	Interpret vector parameters as 1-D	off {on}

表 A-3: Sources ライブラリブロックパラメータ (Continued)

表 A-4: Sinks ライブラリブロックパラメータ

ブロック (Block Type)/ パラメータ	ダイアログポックスプロ ンプト	值
Display (Display)		
Format	Format	{short} long short_e long_e bank
Decimation	Decimation	スカラ {1}
Floating	Floating display	{off} on
SampleTime	Sample time	スカラ (サンプル周期) {-1} またはベクトル [period offset]
Scope (Scope)		
Location	Position of Scope window on screen	ベクトル {[left top right bottom]}
Open	(If Scope open when the model is opened. Cannot set from dialog box)	{off} on
NumInputPorts	Number of Axes	正の整数 > 0
TickLabels	Hide tick labels	{on} off
ZoomMode	(Zoom button initially pressed)	{on} xonly yonly
AxesTitles	Title (on right click axes)	スカラ {auto}
Grid	(for future use)	{on} off
TimeRange	Time range	スカラ {auto}
YMin	Y min	スカラ {-5}
YMax	Y max	スカラ {5}
SaveToWorkspace	Save data to workspace	{off} on
SaveName	Variable name	変数 {ScopeData}
DataFormat	Format	{ 行列 構造体 }

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
LimitMaxRows	Limit rows to last	{on} off
MaxRows	(no label)	スカラ {5000}
Decimation	(Value if Decimation selected)	スカラ {1}
SampleInput	(Toggles with Decimation)	{off} on
SampleTime	(SampleInput value)	スカラ (サンプル周期) {0} またはベクトル [period offset]
Stop Simulation (StopSimulation) (no bloc	k-specific parameters)	
To File (ToFile)		
Filename	Filename	ファイル名 {untitled.mat}
MatrixName	Variable name	变数 {ans}
Decimation	Decimation	スカラ {1}
SampleTime	Sample time	スカラ (サンプル周期) {-1 } またはベクトル [period offset]
To Workspace (ToWorkspace)		
VariableName	Variable name	变数 {simout}
Buffer	Maximum number of rows	スカラ {inf}
Decimation	Decimation	スカラ {1}
SampleTime	Sample time	スカラ (サンプル周期) {-1 } またはベクトル [period offset]
XY Graph (XY scope.) (masked)		

表 A-4: Sinks ライブラリブロックパラメータ (Continued)

表 A-5: Discrete ライブラリブロックパラメータ

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	値
Discrete Filter (DiscreteFilter)		
Numerator	Numerator	ベクトル {[1]}
Denominator	Denominator	ベクトル {[1 2]}
SampleTime	Sample time	スカラ (サンプル周期) {1} またはベクトル [period offset]
Discrete State-Space (DiscreteStateSpace)		
А	А	行列 {1}
В	В	行列 {1}
С	С	行列 {1}
D	D	行列 {1}
X0	Initial conditions	ベクトル {0}
SampleTime	Sample time	スカラ (サンプル周期) {1} またはベクトル [period offset]
Discrete-Time Integrator (DiscreteIntegrat	or)	
IntegratorMethod	Integrator method	{ForwardEuler} BackwardEuler Trapezoidal
ExternalReset	External reset	{none} rising falling either
InitialConditionSource	Initial condition source	{internal} external
InitialCondition	Initial condition	スカラまたはベクトル {0}
LimitOutput	Limit output	{off} on
UpperSaturationLimit	Upper saturation limit	スカラまたはベクトル {inf}
LowerSaturationLimit	Lower saturation limit	スカラまたはベクトル {inf}
ShowSaturationPort	Show saturation port	{off} on
ShowStatePort	Show state port	{off} on

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
SampleTime	Sample time	スカラ (サンプル周期) {1} またはベクトル [period offset]
Discrete Transfer Fcn (DiscreteTransferFc	n)	
Numerator	Numerator	ベクトル {[1]}
Denominator	Denominator	ベクトル {[1 0.5]}
SampleTime	Sample time	スカラ (サンプル周期) {1} またはベクトル [period offset]
Discrete Zero-Pole (DiscreteZeroPole)		
Zeros	Zeros	ベクトル {[1]}
Poles	Poles	ベクトル [0 0.5]
Gain	Gain	スカラ {1}
SampleTime	Sample time	スカラ (サンプル周期) {1} またはベクトル [period offset]
First-Order Hold (First Order Hold) (mask	ed)	
Unit Delay (UnitDelay)		
X0	Initial condition	スカラまたはベクトル {0}
SampleTime	Sample time	スカラ (サンプル周期) {1} またはベクトル [period offset]
Zero-Order Hold (ZeroOrderHold)		
SampleTime	Sample time	スカラ (サンプル周期) {1} またはベクトル [period offset]

表 A-5: Discrete ライブラリブロックパラメータ (Continued)

表 A-6: Continuous ライブラリプロックパラメータ

ブロック (Block Type)/ パラメータ	ダイアログポックスプロ ンプト	值
Derivative (Derivative) (no block-specific parameters)		
Integrator (Integrator)		
ExternalReset	External reset	{none} rising falling either
InitialConditionSource	Initial condition source	{internal} external
InitialCondition	Initial condition	スカラまたはベクトル {0}
LimitOutput	Limit output	{off} on
UpperSaturationLimit	Upper saturation limit	スカラまたはベクトル {inf}
LowerSaturationLimit	Lower saturation limit	スカラまたはベクトル {inf}
ShowSaturationPort	Show saturation port	{off} on
ShowStatePort	Show state port	{off} on
AbsoluteTolerance	Absolute tolerance	スカラ {auto}
Memory (Memory)		
X0	Initial condition	スカラまたはベクトル {0}
InheritSampleTime	Inherit sample time	{off} on
State-Space (StateSpace)		
А	А	行列 {1}
В	В	行列 {1}
С	С	行列 {1}
D	D	行列 {1}
X0	Initial conditions	ベクトル {0}
Transfer Fcn (TransferFcn)		
Numerator	Numerator	ベクトルまたは行列 {[1]}

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
Denominator	Denominator	ベクトル {[1 1]}
Transport Delay (TransportDelay)		
DelayTime	Time delay	スカラまたはベクトル {1}
InitialInput	Initial input	スカラまたはベクトル {0}
BufferSize	Initial buffer size	スカラ {1024}
Variable Transport Delay (VariableTranspo	ortDelay)	
MaximumDelay	Maximum delay	スカラまたはベクトル {10}
InitialInput	Initial input	スカラまたはベクトル {0}
MaximumPoints	Buffer size	スカラ {1024}
Zero-Pole (ZeroPole)		
Zeros	Zeros	ベクトル {[1]}
Poles	Poles	ベクトル {[0-1]}
Gain	Gain	ベクトル {[1]}

表 A-6: Continuous ライブラリブロックパラメータ (Continued)

表 A-7: Math ライブラリブロックパラメータ

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
Abs (Abs) (no block-specific parameters)		
Algebraic Constraint (Algebraic Constraint) (masked)		
Combinatorial Logic (CombinatorialLogic)		
TruthTable	Truth table	行列 {[0 0;0 1;0 1;1 0; 0 1;1 0;1 0;1 1]}
Complex to Magnitude-Angle		

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	値
Complex to Real-Imag		
Dot Product (Dot Product) (masked)		
Gain (Gain)		
Gain	Gain	スカラまたはベクトル {1}
Logical Operator (Logic)		
Operator	Operator	{AND} OR NAND NOR XOR NOT
Inputs	Number of input ports	スカラ {2}
Magnitude-Angle to Complex		
Math Function (Math)		
Operator	Function	{exp} log log10 square sqrt pow reciprocal hypot rem mod
Matrix Gain (Matrix Gain) (masked)		
MinMax (MinMax)		
Function	Function	{min} max
Inputs	Number of input ports	スカラ {1}
Product (Product)		
Inputs	Number of inputs	スカラ {2}
Relational Operator (RelationalOperator)		
Operator	Operator	$== != < {<=} >= >$
Relational Operator (RelationalOperator)		
Operator	Operator	$== != < \{<=\} >= >$
Rounding Function (Rounding)		
Operator	Function	{floor} ceil round fix
Sign (Signum) (no block-specific parameter	ers)	

表 A-7: Math ライブラリブロックパラメータ (Continued)

表 A-7: Math ライプラリプロックパラメータ (Continued)

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
Slider Gain (SliderGain) (masked)		
Sum (Sum)		
Inputs	List of signs	スカラまたは符号リスト {++}
Trigonometric Function (Trigonometry)		
Operator	Function	{sin} cos tan asin acos atan atan2 sinh cosh tanh

表 A-8: Functions and Tables ブロックパラメータ

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
Fcn (Fcn)		
Expr	Expression	表現 {sin(u(1)*exp(2.3*(-u(2))))}
Look-up Table (Lookup)		
InputValues	Vector of input values	ベクトル {[-5:5]}
OutputValues	Vector of output values	ベクトル {tanh([-5:5])}
Look-Up Table (2-D) (Lookup Table (2-D)) (masked)		
RowIndex	Row	ベクトル
ColumnIndex	Column	ベクトル
OutputValues	Table	2-D 行列
MATLAB Fcn (MATLABFcn)		
MATLABFcn	MATLAB function	MATLAB 関数 {sin}
OutputWidth	Output width	スカラまたはベクトル {-1}
S-Function (S-Function)		

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
FunctionName	S-function name	名前 {system}
Parameters	S-function parameters	必要なときの追加パラメータ

表 A-8: Functions and Tables プロックパラメータ (Continued)

表 A-9: Nonlinear ライブラリブロックパラメータ

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
Backlash (Backlash)		
BacklashWidth	Deadband width	スカラまたはベクトル {1}
InitialOutput	Initial output	スカラまたはベクトル {0}
Coulomb & Viscous Friction (Coulombic a	and Viscous Friction) (masked)	
Dead Zone (DeadZone)		
LowerValue	Start of dead zone	スカラまたはベクトル {-0.5}
UpperValue	End of dead zone	スカラまたはベクトル {0.5}
Manual Switch (Manual Switch) (masked)		
Multiport Switch (MultiPortSwitch)		
Inputs	Number of inputs	スカラまたはベクトル {3}
Quantizer (Quantizer)		
QuantizationInterval	Quantization interval	スカラまたはベクトル {0.5}
Rate Limiter (RateLimiter)		
RisingSlewLimit	Rising slew rate	スカラまたはベクトル {1.}
FallingSlewLimit	Falling slew rate	スカラまたはベクトル {-1.}
Relay (Relay)		
OnSwitchValue	Switch on point	スカラまたはベクトル {eps}
OffSwitchValue	Switch off point	スカラまたはベクトル {eps}

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
OnOutputValue	Output when on	スカラまたはベクトル {1}
OffOutputValue	Output when off	スカラまたはベクトル {0}
Saturation (Saturate)		
UpperLimit	Upper limit	スカラまたはベクトル {0.5}
LowerLimit	Lower limit	スカラまたはベクトル {-0.5}
S-Function (S-Function)		
FunctionName	S-function name	名前 {system}
Parameters	S-function parameters	必要なときの追加パラメータ
Sign (Signum) (no block-specific parameter	ers)	
Switch (Switch)		
Threshold	Threshold	スカラまたはベクトル {0}

表 A-9: Nonlinear ライプラリプロックパラメータ (Continued)

表 A-10: Signals & Systems ライブラリブロックパラメータ

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
Bus Selector (BusSelector)		
InputSignals		信号の階層を映す入れ子になった入力信号 のセル配列
Configurable Subsystem (mask)		
Choice	Block choice	文字列
LibraryName	Library name	文字列
Data Store Memory (DataStoreMemory)		
DataStoreName	Data store name	タグ {A}

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	値
InitialValue	Initial value	ベクトル {0}
Data Store Read (DataStoreRead)		
DataStoreName	Data store name	タグ {A}
SampleTime	Sample time	スカラ (サンプル周期) {-1} またはベクトル [period offset]
Data Store Write (DataStoreWrite)		
DataStoreName	Data store name	タグ {A}
SampleTime	Sample time	スカラ (サンプル周期) {-1} またはベクトル [period offset]
Data Type Conversion		
Demux (Demux)		
Outputs	Number of outputs	スカラまたはベクトル {3}
Enable (EnablePort)		
StatesWhenEnabling	States when enabling	{held} reset
ShowOutputPort	Show output port	{off} on
From (From)		
GotoTag	Goto tag	タグ {A}
Goto (Goto)		
GotoTag	Tag	タグ {A}
TagVisibility	Tag visibility	{local} scoped global
Goto Tag Visibility (GotoTagVisibility)		
GotoTag	Goto tag	タグ {A}
Ground (Ground) (no block-specific param	neters)	
Hit Crossing (HitCross)		

表 A-10: Signals & Systems ライブラリブロックパラメータ (Continued)

表 A-10: Signals & Systems ライブラリブロックパラメータ (Continued)

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
HitCrossingOffset	Hit crossing offset	スカラまたはベクトル {0}
HitCrossingDirection	Hit crossing direction	rising falling {either}
ShowOutputPort	Show output port	{on} off
IC (InitialCondition)		
Value	Initial value	スカラまたはベクトル {1}
In (Inport)		
Port	Port number	スカラ {1}
PortWidth	Port width	スカラ {-1}
SampleTime	Sample time	スカラ (サンプル周期) {-1 } またはベクトル [period offset]
Merge		
Model Info (CMBlock) (mask)		
Mux (Mux)		
Inputs	Number of inputs	スカラまたはベクトル {3}
Out (Outport)		
Port	Port number	スカラ {1}
OutputWhenDisabled	Output when disabled	{held} reset
InitialOutput	Initial output	スカラまたはベクトル {0}
Probe (Probe)		
ProbeWidth	Probe width	{on} off
ProbeSampleTime	Probe sample time	{on} off
ProbeCompexSignal	Probe complex signal	{on} off
Subsystem (SubSystem)		

ブロック (Block Type)/ パラメータ	ダイアログボックスプロ ンプト	值
ShowPortLabels	Show/Hide Port Labels Format menu item	{on} off
Terminator (Terminator) (no block-specific	c parameters)	
Trigger (TriggerPort)		
TriggerType	Trigger type	{rising} falling either function-call
ShowOutputPort Show output port		{off} on
Width (Width) (no block-specific parameter	ers)	

表 A-10: Signals & Systems ライプラリプロックパラメータ (Continued)

マスクパラメータ

この節では、マスクされたブロックを記述するパラメータをリストします。つぎ の表は、Mask Editor ダイアログボックスパラメータに対応するマスキングパラ メータのリストです。

表 A-11: マスクパラメータ

パラメータ	説明 / プロンプト	值
Mask	マスクの on、off	{on} off
MaskCallbackString	Mask parameter callbacks	区切り文字列
MaskCallbacks	Mask parameter callbacks	セル配列
MaskDescription	Block description	文字列
MaskDisplay	Drawing commands	表示コマンド
MaskEditorHandle	Mask editor figure handle (for internal use)	handle
MaskEnableString	Mask parameter enable status	区切り文字列
MaskEnables	Mask parameter enable status	文字列のセル配列, 各々 'on' あるいは 'off'
MaskHelp	Block help	文字列
MaskIconFrame	Icon frame (表示は on,、非表示は off)	{on} off
MaskIconOpaque	Icon transparency (不透明は on、透明は off)	{on} off
MaskIconRotate	Icon rotation (回転は on、固定は off)	on {off}
MaskIconUnits	Drawing coordinates	Pixel {Autoscale} Normalized
MaskInitialization	Initialization commands	MATLAB コマンド
MaskNames		
MaskPrompts	Prompt (下記参照)	文字列のセル配列
MaskPromptString	Prompt (下記参照)	区切り文字列
MaskPropertyNameString		
MaskSelfModifiable	Indicates that the block can modify itself.	on {off}

表	A-11	: र	スク	パラ	メー	タ
---	------	-----	----	----	----	---

パラメータ	説明 / プロンプト	值
MaskStyles	Control type (下記参照)	セル配列 {Edit} Checkbox Popup
MaskStyleString	Control type (下記参照)	{Edit} Checkbox Popup
MaskTunableValues	Tunable parameter attributes	文字列のセル配列
MaskTunableValue String	Tunable parameter attributes	区切り文字列
MaskType	Mask type	文字列
MaskValues	Block parameter values (下記参照)	文字列のセル配列
MaskValueString	Block parameter values (下記参照)	区切り文字列
MaskVariables	Variable (下記参照)	文字列
MaskVisibilities	Specifies visibility of parameters	

Mask Editor を使ってマスクされたブロックに対するダイアログボックスパラ メータを作成する場合、つぎの情報を与えます。

- プロンプト。Prompt フィールドに入力します。
- ・ パラメータ値を保持する変数。Variable フィールドに入力します。
- 作成されるフィールドのタイプ。Control type を選択することによって指定します。
- フィールドに入力される値を評価するか文字として保存するかを、 Assignment タイプを選択することによって指定します。

前ページの表に記載してあるマスクパラメータは、ダイアログボックスパラメー タに対して指定される値をつぎの方法で格納します。

- すべてのダイアログボックスパラメータに対する Prompt フィールドの値は、 つぎの例に示すように、個々の値を垂直バー())で区切った文字列として、 MaskPromptString パラメータに格納されます。
 "Slope:|Intercept:"
- すべてのダイアログボックスパラメータに対する Variable フィールドの値は、 個々の割り当てをセミコロンで区切った文字列として、MaskVariables パラ メータに格納されます。連続番号は、どのプロンプトが変数と関連している

かを示します。連続番号に先行する特殊キャラクタは、Assignment タイプを 示します。@はEvaluateで、&はLiteralを意味します。 たとえば、"a=@1;b=&2;"では、最初のパラメータフィールドに入力される値 は変数 a に割り当てられ、割り当ての前に MATLAB によって評価されます。 そして、2番目のフィールドに入力される値は変数 b に割り当てられ、文字と して保存されます。これは、その値がダイアログボックスに入力される文字 列であることを意味しています。

 すべてのダイアログボックスパラメータに対する Control type フィールドの値 は、個々の値をコンマで区切った文字列として、MaskStyleString パラメータに 格納されます。Popup strings の値は、つぎの例に示すように popup タイプの後 に表示されます。

"edit,checkbox,popup(red|blue|green)"

 パラメータ値は、個々の値を垂直バーで区切った文字列として、 MaskValueString マスクパラメータに格納されます。値の順序は、ダイアログ ボックスに表示されるパラメータの順序と同じです。たとえば、つぎのス テートメントは、パラメータフィールドプロンプトに対する値とそれらのパ ラメータに対する値を定義しています。

MaskPromptString "Slope:|Intercept:" MaskValueString "2|5"

モデルファイルフォーマッ ト

モデルファイルの内容								. B-2
Model セクション								. B-3
BlockDefaults $\tau 2 \rightarrow 3$.								. B-3
AnnotationDefaults セクション								. B-3
System セクション								. B-3

モデルファイルの内容

モデルファイルは、モデルを記述するキーワードとパラメータ値の組合わせを含む構造化 ASCII ファイルです。ファイルは、モデルの構成要素を階層的に記述します。

モデルファイルの構造は、つぎのとおりです。 Model { <Model Parameter Name> <Model Parameter Value> ... BlockDefaults { <Block Parameter Name> <Block Parameter Value> ... } AnnotationDefaults { <Annotation Parameter Name> <Annotation Parameter Value> ••• } System { <System Parameter Name> <System Parameter Value> ... Block { <Block Parameter Name> <Block Parameter Value> ... } Line { <Line Parameter Name> <Line Parameter Value> ••• Branch { <Branch Parameter Name> <Branch Parameter Value> ... } } Annotation { <Annotation Parameter Name> <Annotation Parameter Value> ... } } }

モデルファイルは、異なるモデル構成要素を記述するセクションから構成されて います。

- Model セクションは、モデルパラメータを定義します。
- BlockDefaults セクションは、モデル内のブロックに対するデフォルト設定値を 含みます。
- AnnotationDefaults セクションは、モデル内の注釈に対するデフォルト設定値を 含みます。
- System セクションは、モデル内の各システム(最上位システムと各サブシステムを含みます)を記述するパラメータを含みます。各 System セクションには、 ブロック、ライン、および注釈の記述が含まれます。

すべてのモデルパラメータとブロックパラメータについては、付録 A で説明して います。

Model セクション

モデルファイルの最上部にある Model セクションは、モデルレベルのパラメータ に対する値を定義します。これらのパラメータには、モデル名、モデルの最後の 修正に使った Simulink のバージョン、シミュレーションパラメータが含まれま す。

BlockDefaults セクション

BlockDefaults セクションは、シミュレーションパラメータの後にあり、このモデ ル内のブロックパラメータに対するデフォルト値を定義します。これらの値は、 Block セクションで定義されている個々のブロックパラメータで上書きすること ができます。

AnnotationDefaults セクション

AnnotationDefaults セクションは、BlockDefaults セクションの後にあります。このセクションは、モデル内のすべての注釈に対するデフォルトパラメータを定義します。これらのパラメータ値は、set_param コマンドを使って変更することはできません。

System セクション

モデルの最上位システムと各サブシステムは、別々の System セクションに記述 されます。各 System セクションは、システムレベルのパラメータを定義し、シ ステム内の各ブロック、ライン、注釈に対して、Block, Line, Annotation セクションを定義します。分岐点を含む各 Line は、分岐ラインを定義する Branch セクションを含みます。