
speedgoat

Setup Guide

R2020a and earlier

Ver.1.0.1 2021年12月13日

目次

1 概要	1
2 構成	1
2-1 Host computerとTarget computerの接続.....	1
2-2 Target computerとTarget screenの接続.....	1
3 納品物の確認	2
4 カスタマーポータルサイト	2
4-1 アカウントの作成.....	2
4-2 アカウントの有効化.....	2
4-3 Account Home.....	4
4-3-1 Maintenance.....	5
4-3-2 Downloads.....	6
5 開発環境	7
5-1 用意するソフトウェア.....	7
5-1-1 MathWorks製品.....	7
5-1-2 Microsoft製品.....	7
5-1-3 Speedgoat製品.....	8
5-1-4 Xilinx製品.....	8
5-2 Speedgoat製品のインストール.....	9
5-2-1 Speedgoat I/O Blockset.....	9
6 ネットワーク設定	10
6-1 Host computer.....	10
6-2 Target computer.....	11
6-2-1 Kernelの書き換え.....	12
6-2-2 Kernelの書き換え手順.....	15
7 起動	18
7-1 Target computerの起動.....	18
7-2 Target computerとの接続.....	19
7-3 Simulinkモデルのビルド.....	20
7-3-1 オシレータモデル.....	20
7-3-2 サンプルモデル.....	24
8 Control and Instrumentation	25
8-1 MATLAB.....	25
8-1-1 起動.....	25
8-2 Simulink.....	26
8-3 Simulink Real-time Explorer.....	27
8-3-1 リアルタイムアプリケーションのロード.....	28
8-3-2 信号の表示.....	29
8-3-3 パラメータ設定.....	30
8-3-4 Host scopeによる信号の表示.....	31
8-3-5 Target scopeによる信号の表示.....	34
8-3-6 File scopeによる信号の保存.....	35
8-4 App Designer.....	37
8-4-1 起動.....	37

8-4-2 GUIの配置	39
8-4-3 Target computerとの接続	40
8-4-4 モデルのビルド	42
8-4-5 リアルタイムアプリケーションのロード	44
8-4-6 リアルタイムアプリケーションの実行	45
8-4-7 信号の表示	46
8-4-8 パラメータの設定	50

1 概要

本書は、Speedgoat 社のリアルタイムターゲットマシンでサンプルモデルを実行するまでの手順について説明します。

2 構成



Host computer	Target computer で実行するリアルタイムアプリケーションを作成します。リアルタイムアプリケーションは Simulink モデルから生成されます。
Target computer	Speedgoat 製品のリアルタイムターゲットマシンです。
Target screen	Target computer の BIOS 設定や、OS のブート画面、リアルタイムアプリケーションの実行状況を表示します。

2-1 Host computer と Target computer の接続

付属の Ethernet ケーブルで接続します。

Host computer 側は Ethernet 通信が可能な RJ-45 コネクタを使用します。

Target computer 側は host link と記載された RJ-45 コネクタを使用します。

Mobile Target Machine を参考例として示します。



2-2 Target computer と Target screen の接続

使用する Target computer の種類によって使用できるケーブルが異なります。

Target computer の解像度については、それぞれの User Manual をご参照ください。

Performance	HDMI, DVI-I, VGA
Mobile	DVI-I, DisplayPort
Baseline	DisplayPort
Unit	DisplayPort

3 納品物の確認

Speedgoat 製品はターゲットマシンやケーブル、端子台といったハードウェアとライブラリやドキュメントといったソフトウェアで構成されます。

ハードウェアについては弊社からご注文いただいた内容の Speedgoat 製品を納品します。

納品物には納品リストが含まれていますので、内訳はそちらをご確認ください。

ソフトウェアについては Speedgoat 社の HP にあるカスタマーポータルサイトよりダウンロードして入手します。

カスタマーポータルサイトについては後述します。

4 カスタマーポータルサイト

カスタマーポータルサイトでは Speedgoat I/O Blockset やサンプルモデルといったソフトウェア、それらの取扱説明書のダウンロードや購入した製品ハードウェアの保守期間やソフトウェアメンテナンス期間を閲覧できるサービスを提供します。

<https://www.speedgoat.com/extranet#/Login>

4-1 アカウントの作成

カスタマーポータルサイトを利用するためには Speedgoat 社へアカウント作成の依頼をする必要があります。

ただし、注文いただいたユーザのアカウントは納品前後に作成されております。

それ以外のユーザのアカウントを作成する場合は、弊社のサポートや営業、もしくはカスタマーポータルサイトで依頼することになります。

4-2 アカウントの有効化

アカウント作成が完了後、それを有効化することでカスタマーポータルサイトへログインすることができます。

有効化するためには下記 URL へアクセスしてください。

<https://www.speedgoat.com/extranet#/Login/ActivateAccount>

アクセスすると下図のような画面が表示されますので、アカウント作成時に連絡しているメールアドレスを入力します。

Activate your Account

Enter your email address and click Submit. We will send you an email that includes a link to create your password.

Email Address *

Submit

メールアドレスを入力すると、そのメールアドレス宛に下記内容のメールが届きます。

Please create your password by clicking here: [Activate your Account](#)

Please note that this token/link is only valid for 30 minutes. If your token has expired, please request a new one on www.speedgoat.ch/login

Your Speedgoat Team

メールに記載されたリンク「Activate your Account」を選択すると、パスワードを設定する画面が表示されます。

Set your Password

new Password *

Passwords must be at least 8 characters long including letter, numerical digit and a special character

Confirm Password *

Submit

パスワードの設定が完了後、メールの「www.speedgoat.ch/login」を選択して下図のログイン画面を開きます。

Customer Log In

The customer area gives access to:

- Details of your purchased systems
- Renewal of hardware warranty and software maintenance plans
- Download the latest drivers, software and documentation

Email Address *

Password *

Login









[Forgot your password?](#)

Don't have a Speedgoat Account? [Create an account](#)

ログイン画面でメールアドレスとパスワードを入力すると、カスタマーポータルへログインできます。

4-3 Account Home

ログイン後に表示される Account Home 画面が下図になります。

 <p>Maintenance View and renew your maintenance</p>	 <p>Downloads Download drivers, software and documentation</p>
 <p>Learning Tutorials Introduction to Simulink Real-Time</p>	 <p>Documentation Documentation for the Speedgoat Library for Simulink Real-Time and HDL Coder</p>
 <p>Your Company Profile Edit your company's address</p>	 <p>My Profile Edit your name and change password</p>
 <p>Users Add, view and edit the accounts of your company's users</p>	 <p>Log out Log out of the customer area</p>

Maintenance	ハードウェアの保守期間やソフトウェアメンテナンス期間の確認ができます。
Downloads	ライブラリやサンプルモデル、ドキュメントのダウンロードができます。
Learning Tutorials	基本的な操作方法を紹介した動画が閲覧できます。
Documentation	Speedgoat I/O Blockset などの説明が記載されています。
Your Company Profile	アカウントユーザが所属する組織情報が記載されています。
My Profile	アカウントユーザの情報が記載されています。
Users	組織に所属しているユーザー一覧が記載されています。
Log out	ポータルサイトからログアウトします。

4-3-1 Maintenance

Account HomeでMaintenanceを選択すると表示されます。

購入した製品の情報が表示されます。

Maintenance

Mobile real-time target machine - serial number: 3035 >

Performance real-time target machine - serial number: 3036 >

Baseline real-time target machine - S - serial number: 3612 >

Performance real-time target machine - serial number: 4343 ▾

Main user: Souta Kawashima

Description	Item ID	Hardware Warranty expiry date	Software Maintenance expiry date
System	109200 109006 109211	01/Apr/2020	01/Apr/2028
IO322	2A322	01/Apr/2020	01/Apr/2028
IO75X	2A75X	01/Aug/2022	01/Aug/2021

Baseline real-time target machine - M - serial number: 4494 >

Components >

Maintenance Renewal

Request a quote for the renewal of all expired, or soon to expire, target machine and components. Maintaining your subscription ensures that your systems are fully compatible with future MathWorks software releases and enables access to technical support.

Please choose a Renewal Type *

1 Year Renewal

2 Years Renewal

3 Years Renewal

Renewal until end of 2022

Custom (specify below)

Please add any special instructions here

[Request Maintenance Quote](#)

Hardware Warranty expiry date ハードウェアの保守期間です。

Software Maintenance expiry date ソフトウェアのメンテナンス期間です。

4-3-2 Downloads

Account HomeでDownloadsを選択すると表示されます。

Downloads

Speedgoat I/O Blockset

Installation instructions:

1. Download the library for your MATLAB release
2. Extract the downloaded ZIP file into a temporary folder
3. In MATLAB, navigate to the extracted folder which contains speedgoat_setup.p (do not add any folder to the MATLAB path)
4. Right click on the speedgoat_setup.p and select Run
5. Follow the instructions
6. Restart MATLAB

[Important Note: Migration of existing hardware to new R2020b release](#)

[Speedgoat I/O Blockset for R2018b \(v9.2.0\)](#)

[Speedgoat I/O Blockset for R2019a \(v9.2.0\)](#)

[Speedgoat I/O Blockset for R2019b \(v9.2.0\)](#)

[Speedgoat I/O Blockset for R2020a \(v9.2.0\)](#)

[Speedgoat I/O Blockset for R2020b \(v9.2.0\)](#)

[Speedgoat I/O Blockset for R2021a \(v9.2.0\)](#)

[Speedgoat I/O Blockset Release Notes](#)

主要なソフトウェアとドキュメントについて下表に記載します。

Speedgoat I/O Blockset	Speedgoat の I/O ライブラリです。
Configuration and Testing Information	サンプルモデルの説明書です。
Simulink Test Model	サンプルモデルです。
Custom Implementation	FPGA モジュールの bitstream です。
HDL Coder Integration Package	HDL Coder 用 Speedgoat のライブラリです。

※ソフトウェアやドキュメントの名称は、通知なく変更される場合がございますのでご了承ください。

※ソフトウェアの中にはオプション品も含まれ、購入していないと表示されないものがあります。

ソフトウェアメンテナンス期間内の間は最新の Speedgoat I/O Blockset と HDL Coder Integration Package を提供しております。

5 開発環境

5-1 用意するソフトウェア

5-1-1 MathWorks 製品

Simulinkモデルでアプリケーションを作成するために必要なMathWorks製品ソフトウェアを下表に示します。

MathWorks製品
MATLAB
Simulink
MATLAB Coder
Simulink Coder
Simulink Real-time
HDL Coder*
Fixed Point Designer*

*HDL Coder Integration Package使用時のみ必要

これらの製品は MathWorks 社からインストーラをダウンロードし、インストールします。

5-1-2 Microsoft 製品

Simulink モデルでアプリケーションを作成するためには Microsoft 社の Visual Studio (コンパイラ) が必要となります。

また MATLAB のバージョンによって必要な Visual Studio のバージョンも異なります。

下記 URL の「Supported Compilers」を選択すると、各 MATLAB バージョンに必要な Visual Studio の情報が表示されます。

<https://jp.mathworks.com/support/requirements/previous-releases.html>

使用する MathWorks 製品に対応した Visual Studio をインストールします。

MATLAB Product Family – Release 2020a									
Compiler	MATLAB	MATLAB Coder	GPU Coder	SimBiology	Fixed Point Designer	HDL Coder	HDL Verifier	Audio Toolbox	ROS Toolbox
	For MEX-file compilation, loadlibrary , and external usage of MATLAB Engine and MAT-file APIs	For all features	For all features	For accelerated computation	For accelerated computation	For accelerated testbench simulation	For DPI and TLM component generation	For validating and generating audio plugins	For ROS 2 custom messages and code generation
MinGW 6.3 C/C++ (Distributor: mingw-w64) Download Now Available at no charge	✓	✓		✓	✓	✓	✓		
Microsoft Visual C++ 2019 product family	✓ Update 3	✓ Update 5	✓ Update 5	✓ Update 5				✓ Update 5	
Microsoft Visual C++ 2017 product family ¹¹	✓	✓	✓	✓	✓	✓	✓	✓	✓
Microsoft Visual C++ 2015 Professional ¹⁰	✓	✓	✓	✓	✓	✓	✓	✓ ¹	

5-1-3 Speedgoat 製品

Simulinkモデルでアプリケーションを作成する場合は下記ふたつのソフトウェアをカスタマーポータルサイトよりダウンロードします。

どちらもMATLABのバージョンごとに用意されております。

- Speedgoat I/O Blockset
- Speedgoat HDL Coder Integration Package (HDL Coder使用時のみ必要)

※Speedgoat I/O Blocksetは最新Verから数えて6リリース前のMALTABまでをサポートします。

5-1-4 Xilinx 製品

HDL Coderを使用する場合はXilinxの統合開発環境Vivadoが必要となります。

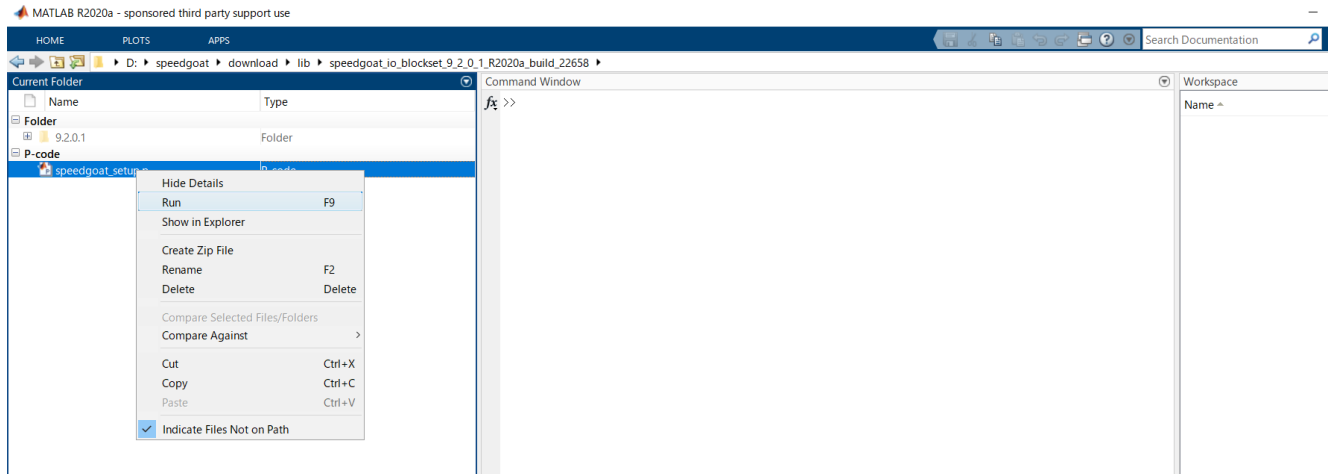
詳細は下記URLをご参照ください。

https://www.speedgoat.com/help/hdlcoder/page/refentry_workflow

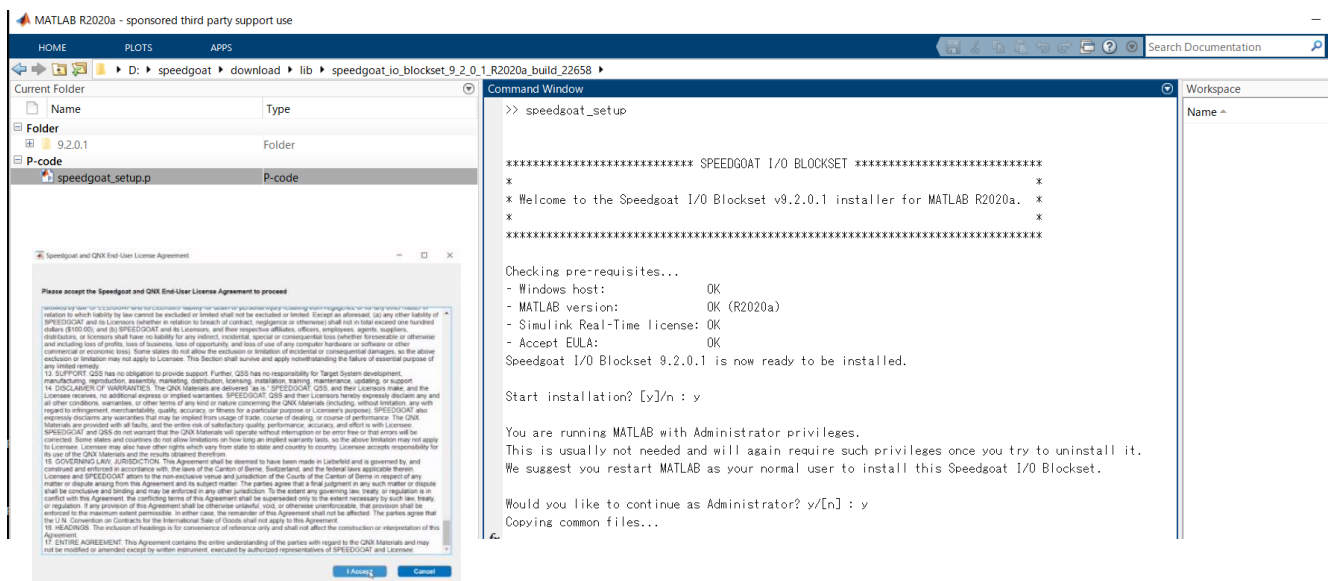
5-2 Speedgoat 製品のインストール

5-2-1 Speedgoat I/O Blockset

カスタマーポータルからダウンロードした Speedgoat I/O Blockset は zip ファイルとなります。それを任意のフォルダに解凍し、そのフォルダを MATLAB の Current Folder にします。解凍したフォルダに格納されている `speedgoat_setup.p` を実行します。



インストールが開始されますので、表示される使用許諾契約をよく読み、許諾を選択します。

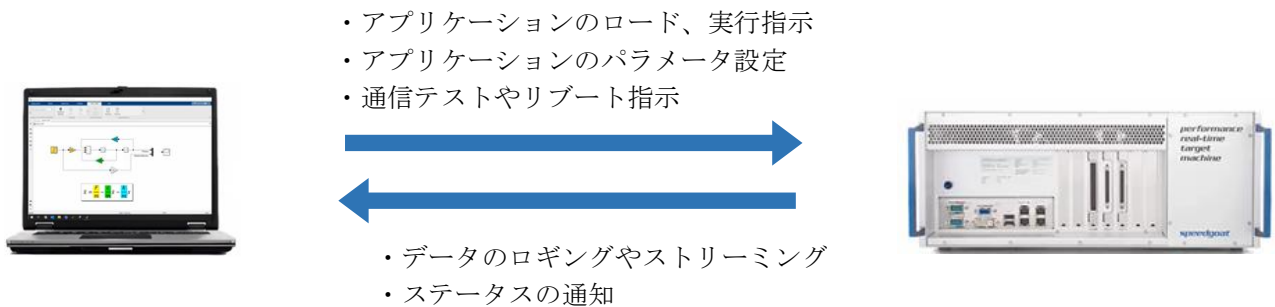


問題なければ、そのまま Speedgoat I/O Blockset のインストールは完了します。

MATLAB の Command Window に `speedgoat.version` と入力すると、現在インストールされているバージョンを確認できます。

6 ネットワーク設定

Host computer と Target computer で Ethernet 通信するためにネットワークの設定をする必要があります。



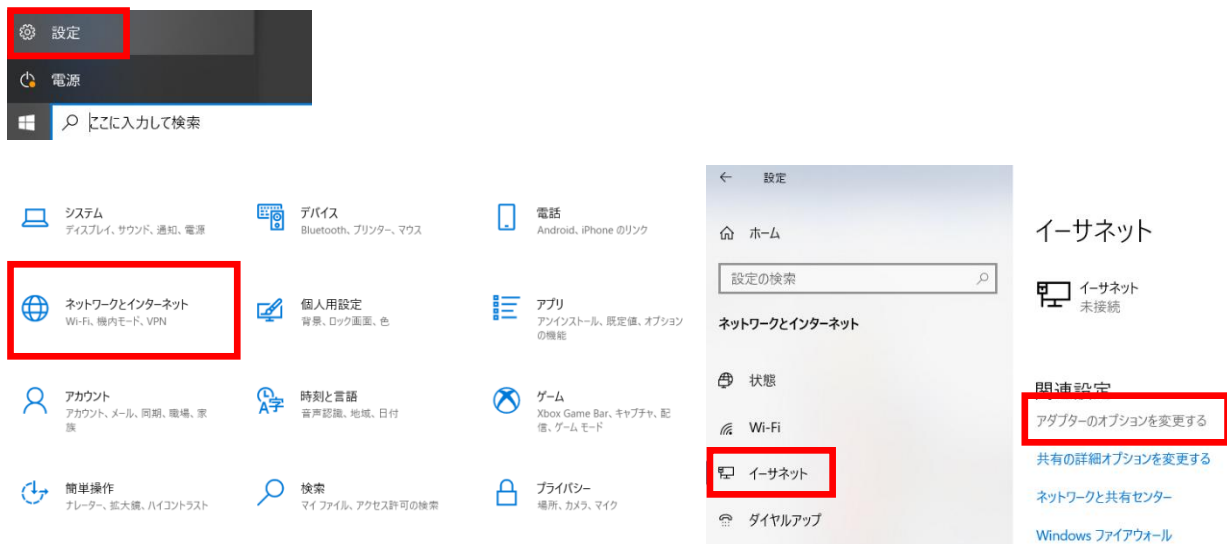
Target computer の出荷時のネットワーク設定は以下のとおりです。

IP Address	192.168.7.1
Subnet mask	255.255.255.0

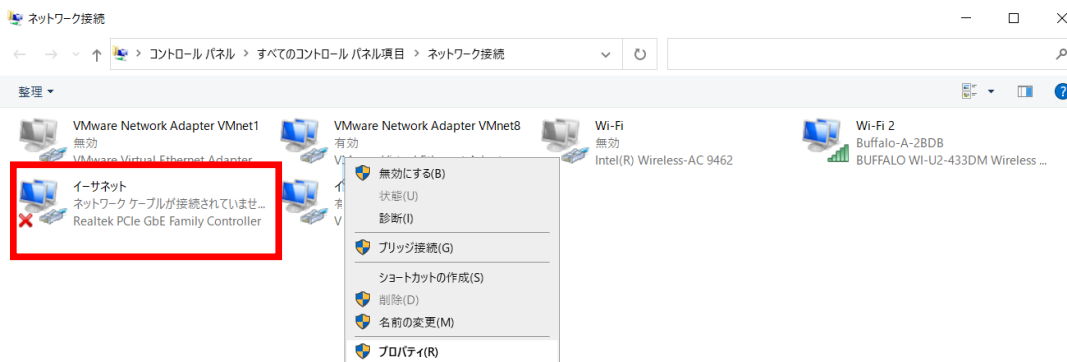
6-1 Host computer

下記の項目から Host computer の設定を変更することができます。

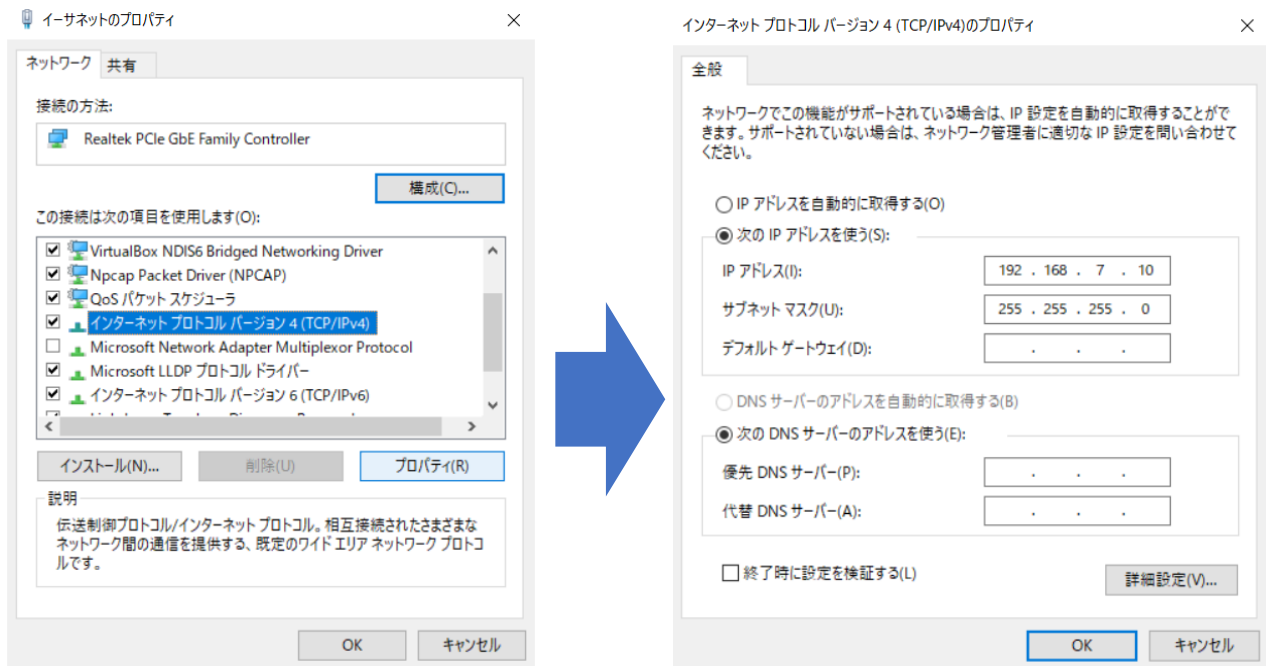
PC の設定 > ネットワークとインターネット > イーサネット > アダプターのオプションを変更する



使用するネットワークカード（ネットワークアダプタ）を右クリックし、プロパティを選択します。下図の例では最も左下の項目が使用するネットワークカードとなります。



インターネットプロトコルバージョン 4(TCP/IPv4)をクリックした後、プロパティをクリックします。
 下図はデフォルト設定の Target computer と接続できる設定となります。



IP Address	192.168.7.10
Subnet mask	255.255.255.0

6-2 Target computer

Target computer のネットワーク設定は、後述する Kernel の書き換えで変更することができます。
 Target computer を起動し、ブートすると Kernel などのバージョン情報が表示されます。

6-2-1 Kernel の書き換え

Host Computer と Target Computer で MATLAB や Speedgoat I/O Blockset のバージョンが異なると、うまく通信できません。この場合 Target Computer の Kernel を書き換える必要があります。

以下に Kernel の書き換え方法について説明します。

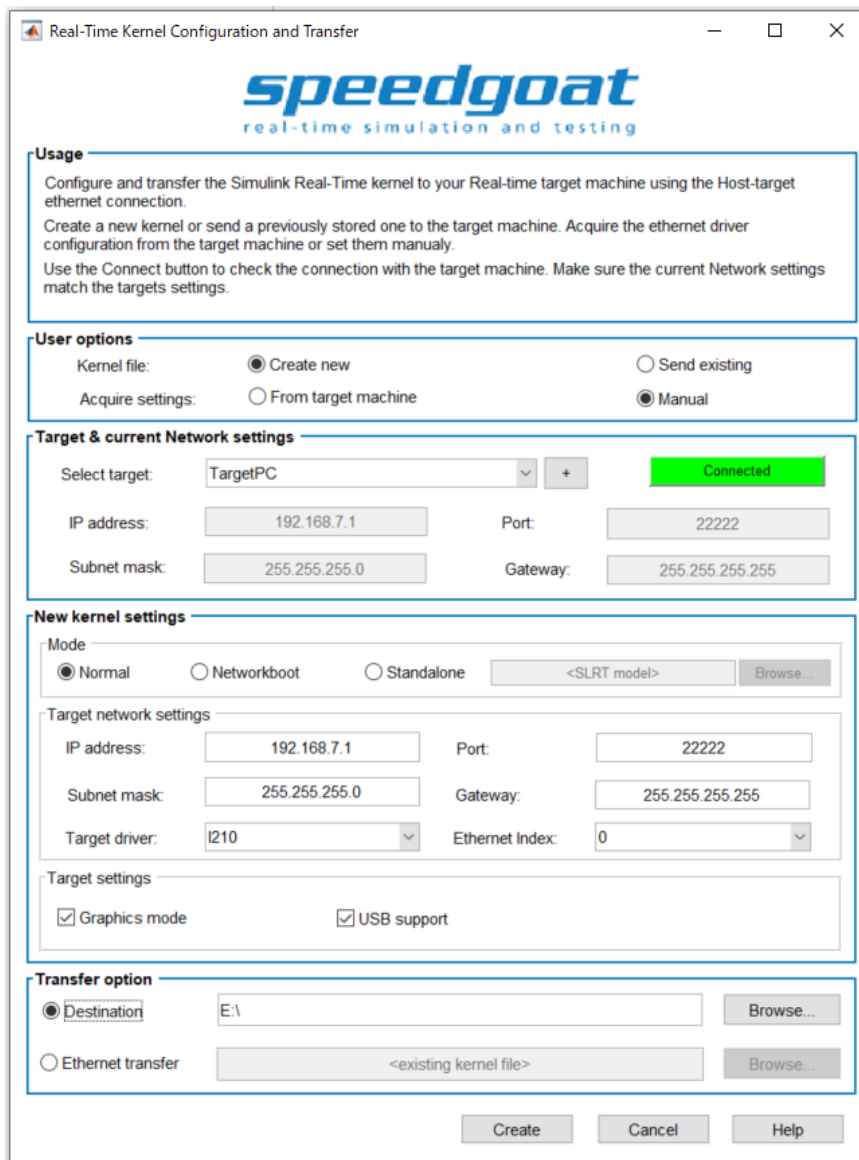
※Target Computer のカーネルバージョンが R2020b 以降の場合は、Target Computer を R2020a 以前のカーネルを読み込める状態に設定する必要があります。

Target Computer のカーネルバージョンの確認方法は、Target Computer 起動後の画面から確認可能です。

R2020a 以前画面表示	R2020b 以降画面表示

「6-2-2 Kernel の書き換え手順」も併せてご参照ください。

MATLAB の Command Window 上で speedgoat.kernelTransfer と入力します。
入力すると、下図のような画面が表示されます。



Kernel を書き換える方法はいくつかありますが、今回は最も基本的な USB メモリによる Kernel の書き換えについて説明します。

- USB メモリを Host Computer へ挿入します。
- Transfer option の欄で Destination をチェックし、認識した USB メモリのフォルダパスを入力します。
- Create をクリックし、USB メモリの最上フォルダ (E:\ 等) へ Kernel が作成されることを確認します。
- USB メモリをターゲットマシンへ挿入し、電源を ON にすると Kernel がロードされます。
- Kernel のロードを確認後、電源を OFF にし、USB メモリを抜去後に電源を ON にします。
- 新しくロードした Kernel でブートすることを確認します。

※設定項目については次頁で説明します。

Usage Options	作成する Kernel の情報を新規作成するか、別の方法で作成するのか選択します。基本的には新規作成をすることをお勧めしますので、” Create new” と “Manual” を指定します。																												
Target & current Network settings	USB メモリではなくネットワーク通信で書き換える場合、設定します。設定する場合は Target Computer のネットワーク設定を入力します。																												
New kernel settings	Target computer へ書き込む Kernel の設定を入力します。 <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th colspan="2">Mode</th> </tr> <tr> <td>Normal</td> <td>後述する Transfer option で選択した方法で Kernel を書き換えます。本紙の方法ではこれを選択します。</td> </tr> <tr> <td>Networkboot</td> <td>Host computer のネットワーク設定を変更できない環境で、ネットワークから Kernel を書き換えるときに選択します。</td> </tr> <tr> <td>Standalone</td> <td>基本的には Normal と同様ですが、Kernel とともに *1 リアルタイムアプリケーション *1 を書き込みます。書き込まれたリアルタイムアプリケーションは Target computer 起動時に自動で実行されます。</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th colspan="2">Target network settings</th> </tr> <tr> <td>IP address</td> <td>Target computer に設定する IP アドレスを入力します。</td> </tr> <tr> <td>Port</td> <td>Target computer に設定する Port 番号を入力します。特に指定がなければ 22222 と入力します。</td> </tr> <tr> <td>Subnet mask</td> <td>Target computer に設定する Subnet mask を入力します。</td> </tr> <tr> <td>Gateway</td> <td>Target computer に設定する Gateway を入力します。</td> </tr> <tr> <td>Target driver</td> <td>Target computer に設定する Target driver を入力します。この項目は Target computer の種類によって決まります。</td> </tr> <tr> <td>Ethernet Index</td> <td>Target computer に設定する Ethernet Index を入力します。この項目は Target computer の種類によって決まります。</td> </tr> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th colspan="2">Target network settings</th> </tr> <tr> <td>Graphics mode</td> <td>Target Screen 上の表示を変更します。チェックを外すことで、グラフ表示などができなくなりますが、処理が若干軽くなります。</td> </tr> <tr> <td>USB support</td> <td>チェックを外すことで USB Web カメラなどの USB 機器が使用できなくなりますが、処理が若干軽くなります。</td> </tr> </table>	Mode		Normal	後述する Transfer option で選択した方法で Kernel を書き換えます。本紙の方法ではこれを選択します。	Networkboot	Host computer のネットワーク設定を変更できない環境で、ネットワークから Kernel を書き換えるときに選択します。	Standalone	基本的には Normal と同様ですが、Kernel とともに *1 リアルタイムアプリケーション *1 を書き込みます。書き込まれたリアルタイムアプリケーションは Target computer 起動時に自動で実行されます。	Target network settings		IP address	Target computer に設定する IP アドレスを入力します。	Port	Target computer に設定する Port 番号を入力します。特に指定がなければ 22222 と入力します。	Subnet mask	Target computer に設定する Subnet mask を入力します。	Gateway	Target computer に設定する Gateway を入力します。	Target driver	Target computer に設定する Target driver を入力します。この項目は Target computer の種類によって決まります。	Ethernet Index	Target computer に設定する Ethernet Index を入力します。この項目は Target computer の種類によって決まります。	Target network settings		Graphics mode	Target Screen 上の表示を変更します。チェックを外すことで、グラフ表示などができなくなりますが、処理が若干軽くなります。	USB support	チェックを外すことで USB Web カメラなどの USB 機器が使用できなくなりますが、処理が若干軽くなります。
Mode																													
Normal	後述する Transfer option で選択した方法で Kernel を書き換えます。本紙の方法ではこれを選択します。																												
Networkboot	Host computer のネットワーク設定を変更できない環境で、ネットワークから Kernel を書き換えるときに選択します。																												
Standalone	基本的には Normal と同様ですが、Kernel とともに *1 リアルタイムアプリケーション *1 を書き込みます。書き込まれたリアルタイムアプリケーションは Target computer 起動時に自動で実行されます。																												
Target network settings																													
IP address	Target computer に設定する IP アドレスを入力します。																												
Port	Target computer に設定する Port 番号を入力します。特に指定がなければ 22222 と入力します。																												
Subnet mask	Target computer に設定する Subnet mask を入力します。																												
Gateway	Target computer に設定する Gateway を入力します。																												
Target driver	Target computer に設定する Target driver を入力します。この項目は Target computer の種類によって決まります。																												
Ethernet Index	Target computer に設定する Ethernet Index を入力します。この項目は Target computer の種類によって決まります。																												
Target network settings																													
Graphics mode	Target Screen 上の表示を変更します。チェックを外すことで、グラフ表示などができなくなりますが、処理が若干軽くなります。																												
USB support	チェックを外すことで USB Web カメラなどの USB 機器が使用できなくなりますが、処理が若干軽くなります。																												
Transfer option	作成した Kernel の保存場所を指定します。Destination を選択した場合、Host computer 上に Kernel を保存します。通常は USB メモリに保存し、それを使用して Kernel の書き換えを行います。Ethernet transfer を選択した場合、ネットワーク経由で Target computer へ Kernel を直接書き込みます。この場合、Target & current Network settings であらかじめ Target computer とネットワーク接続している必要があります。																												

*1 リアルタイムアプリケーションは Target computer で実行されるファイルです。

Target driver と Ethernet Index は Target computer の種類によって異なりますので、下表を参考に適切な値を設定してください。

Target driver	Performance (4.2GHz)	I217/9
	Performance (3.5GHz), Mobile	Intel 8254x
	Baseline系、Unit	I210
Ethernet Index	Performance (4.2GHz)	0
	Performance (3.5GHz), Mobile	0
	Unit	0
	Baseline系	3

※Ethernet Index は I0791 などネットワーク通信系の 10 モジュールが搭載されている場合、変動することがあります。正しい値は Target computer 本体にラベリングされていますので、それを確認してください。

6-2-2 Kernel の書き換え手順

USB メモリによる Kernel の書き換え手順は下記となります。

Target Computer の Kernel が 2020b 以降の場合は手順 1 から実施します。

Target Computer の Kernel が 2020a 以前の場合は手順 5 から実施します。

1. speedgoat Target Software Migration とプリントされている USB メモリ (以後 speedgoat USB メモリ) を使用します。

speedgoat ターゲットマシンご購入時の付属品です。

※kernelTransfer でカーネルファイルを格納する USB メモリとは別のものになります。



speedgoat USB メモリには 2020b 以降の Kernel の情報が入っています。

speedgoat USB で R2020a 以前のバージョンを読み込めるように設定するため、speedgoat USB をターゲットマシンでブート起動します。

2. ブートに成功すると、下図画面が表示されるので、Enter キーを押します。

```
Starting some common services ...
Starting network service ...
Starting USB host ...
Starting devb-umass ...
Starting input services ...
Starting console ...
Setting up SSD for QNX ...
Fri Mar 05 18:30:39 GMT 2021
Starting Hard Disk and USB drivers ...
Starting target machine authentication process ...
Successfully authenticated this target machine

   OS          Start      End      Number      Size      Boot
  name        type      Cylinder Cylinder  Cylinders  Blocks
1.  FAT32      12         512      229588      469133312 229069 MB *
2.  FAT32      12        229581    241663      12083      24745984 12083 MB
3.  -----
4.  -----

This disk is going to be formatted. All partitions are going to be deleted.
=====
To format this SSD drive, Press Enter
To exit, Press Ctrl+C
```

3. 下図画面が表示されるので、「3」を入力して Enter キーを押します。

```
=====
Choose one of the following installation options

[1] Set Up a Dual-Boot System
    Install both operating systems in dual-boot mode:
    the QNX-based 64-bit Simulink Real-Time for MATLAB R2020b and later
    with the Speedgoat target software and
    the 32-bit Simulink Real-Time for MATLAB R2020a and earlier.
    NOTE: Dual-boot mode does not have any negative impact on real-time
    performance or compatibility.

[2] Update to R2020b and later
    Install the QNX-based 64-bit Simulink Real-Time
    for MATLAB R2020b and later with the Speedgoat target software.

[3] Revert to R2020a and earlier
    Install the 32-bit Simulink Real-Time for MATLAB R2020a and earlier.

=====
```

4. 下図画面が表示されるので、Speedgoat USB メモリを抜きます。

```
=====
3
Reverting to R2020a and earlier
Formatting disk...
Setting up the image file system ...

Format complete: FAT32 (4096-byte clusters), 234189392 kB available.
Format complete: FAT32 (4096-byte clusters), 12348856 kB available.
Finalizing the installation ...
1+0 records in
1+0 records out
78+1 records in
78+1 records out
=====
=====

Revert Installation Complete!

Remove the USB drive and press Enter to reboot.

=====
=====
```

5. 「6-2-1 Kernel の書き換え」で作成した 2020a 以前のカーネル OS が入った USB メモリを挿入し、再起動してターゲットマシンでブート起動します。

6. ブート後、下図画面が表示された場合は Enter キーを押します。

```
Press F1-F4 to select drive or select partition ? 1
Remove disks or other media.
Press any key to restart
```

7. 下図画面が表示されたら、USB メモリを取り外し、ターゲットマシンを再起動します。

```
baseline
real-time
target
machine

speedgoat

Simulink Real-Time kernel has been successfully transferred
from USB memory device to solid state drive
Please remove the USB memory device and reboot.
C:\>_
```

8. 再起動後、画面に Simulink Real-Time バージョンと IP アドレス等のネットワーク情報が表示されれば完了です。



※画面に Simulink Real-Time バージョンのみ表示される場合は、「6-2-1 Kernel の書き換え」の Target driver または Ethernet Index の設定に誤りがある場合があります。

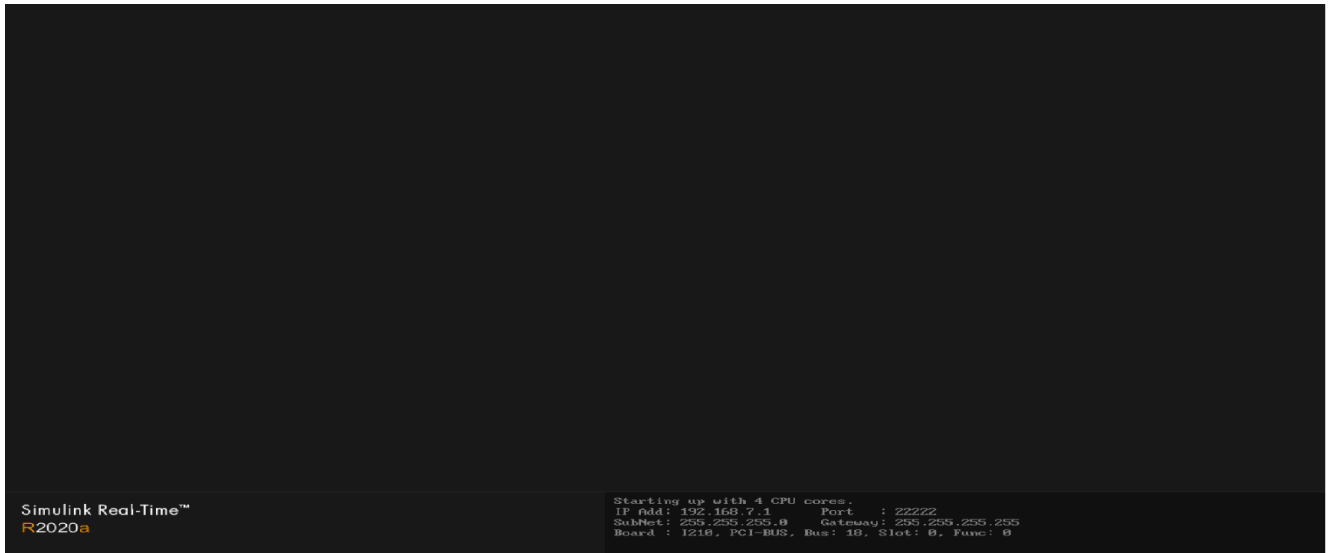
7 起動

7-1 Target computer の起動

Target computer へ電源を投入します。

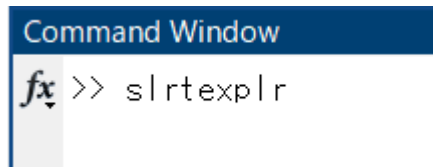
Target computer は起動すると、BIOS > OS の順でブートします。

OS の Boot に成功すると、Target screen 上にネットワークなどの各種情報が表示されます。

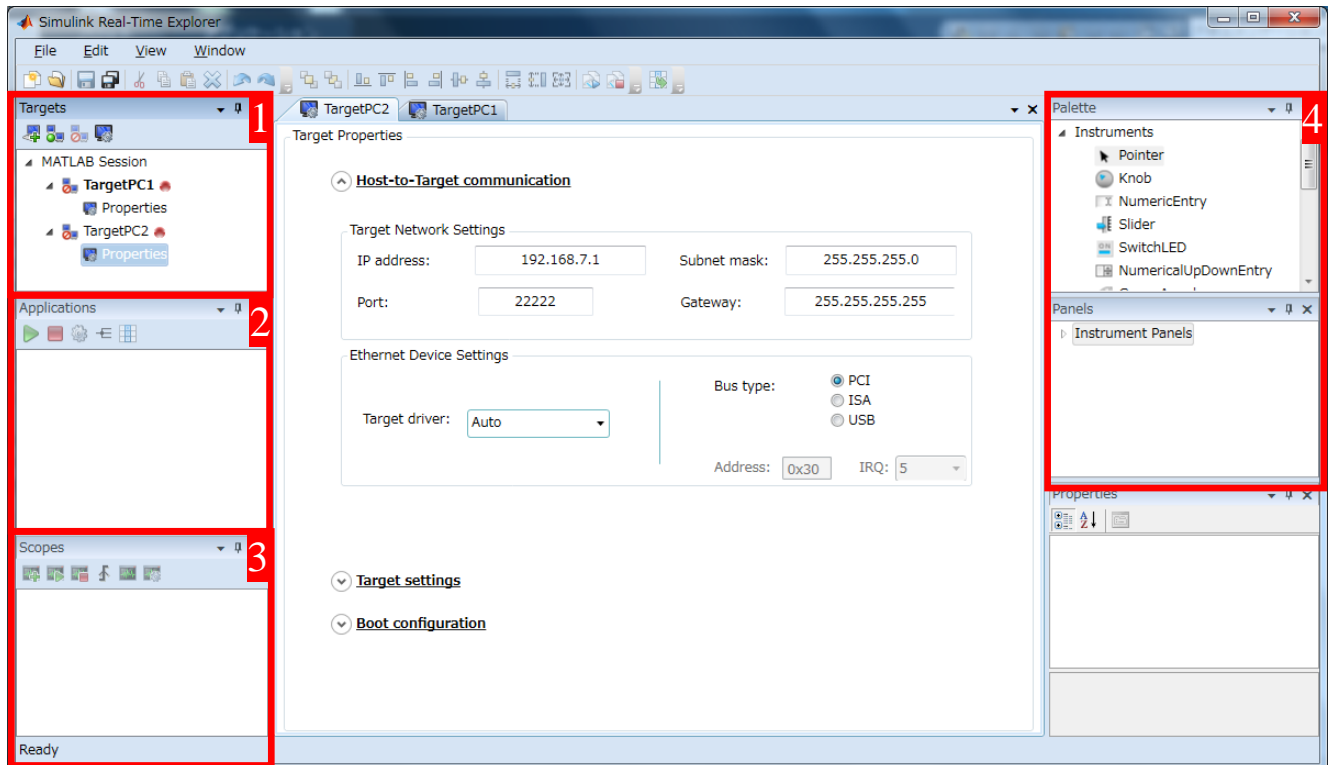


7-2 Target computer との接続

Host computer から Target computer へ接続確認を行います。
 接続確認は MATLAB の Simulink Real-time Explorer から行います。
 MATLAB の Command Window で slrtexplr と入力します。



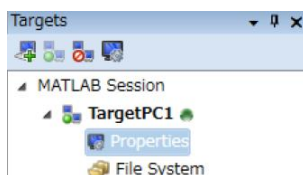
・ Simulink Real-time Explorer の画面



- 1 接続する Target computer の設定や接続後の Target computer に対してのリアルタイムアプリケーションのロードや Target computer のファイルシステムへのアクセスなどが行えます。
- 2 Target computer へロードしたリアルタイムアプリケーションの実行や停止、パラメータや信号の確認や変更を行うことができます。
- 3 Simulink モデル上の信号をグラフ表示することができます。
- 4 用意されたパーツを配置し、GUI を作成することができます。作成した GUI に Simulink モデル上のパラメータや信号をリンクさせることで値の表示や変更が可能です。



や をクリックすることで選択中の Target computer (下図では TargetPC1) と接続を試みます。



接続に成功すると、Target computer のボタン が へ変化します。

7-3 Simulink モデルのビルド

カスタマーポータルサイトからダウンロードしたサンプルモデルをビルドし、リアルタイムアプリケーションを生成します。

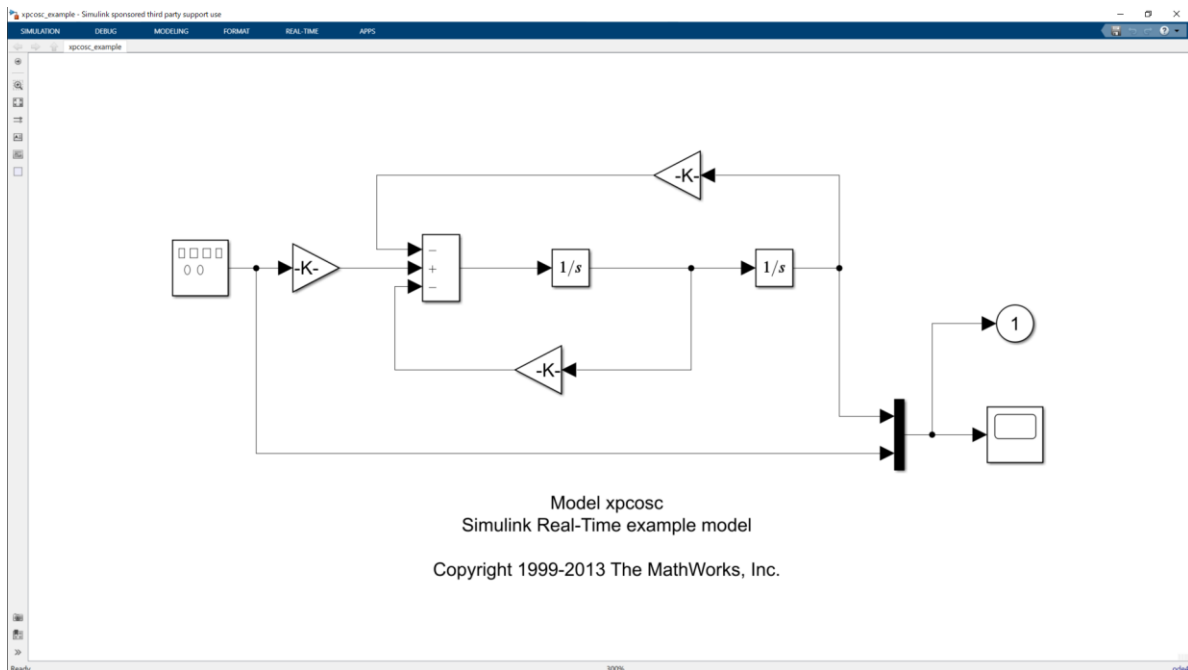
本項では Simulink I/O Blockset を使用していないオシレータモデルをビルドする方法を説明します。

7-3-1 オシレータモデル

MATLAB の Command Window で `xpcosc` と入力します。

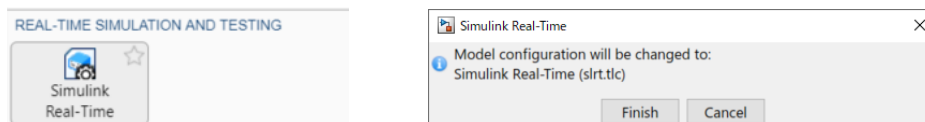
入力すると Simulink 上でオシレータモデル（モデル名：`xpcosc`）が開かれます。

この `xpcosc` を別名保存して、使用してください。

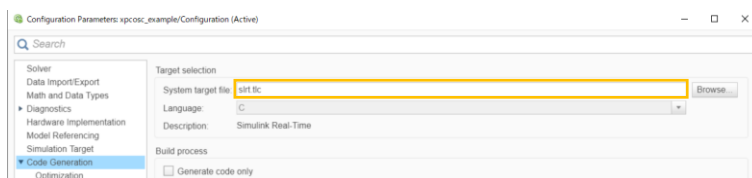


Simulation > Run をクリックすると Host computer 上でオシレータモデルをシミュレーション実行します。

次に APPS > Simulink Real-time をクリックすると、モデルが Simulink Real-time 用のテンプレートファイルを使用する設定になります。



Configuraion Parameters で変更されたテンプレートファイルを確認することができます。



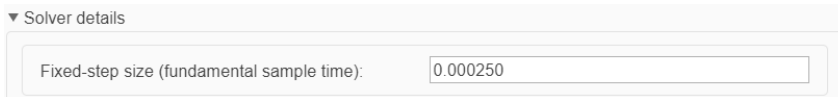
※ただし `xpcosc` の初期設定で Simlink Real-time 用の設定となっております。

Configuraion Parameters の Stop time を inf にした後ビルドすることで、Host computer から停止を指令するまでリアルタイムアプリケーションを実行します。この設定を行った前提で、以降の説明を行います。



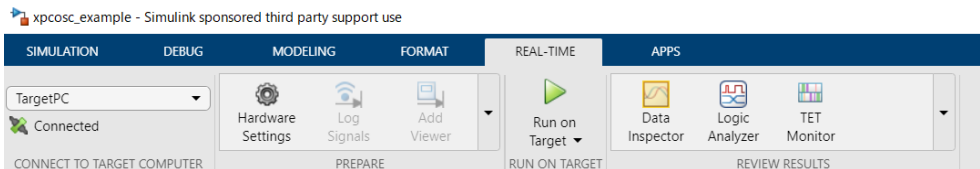
※Configuraion Parameters は  ボタンを押すことで表示されます。

この画面ではモデルの動作周期なども変更することができます。

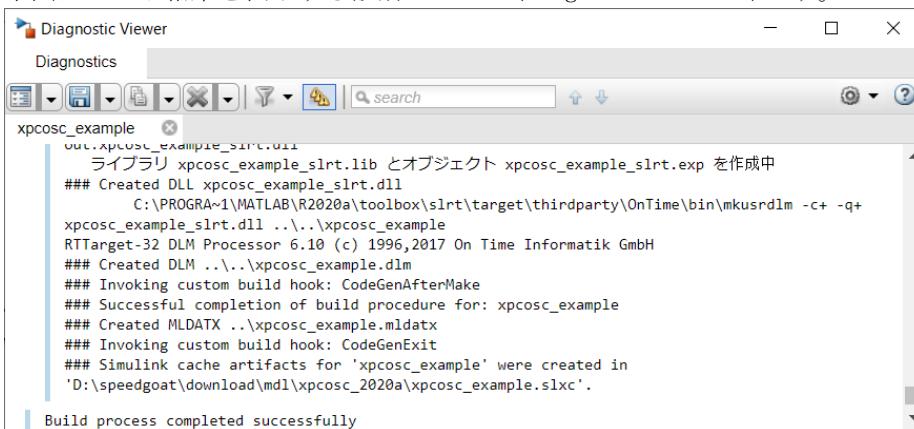


Run on Target をクリックすると、Simulink モデルをビルドし、Target computer で実行することができるリアルタイムアプリケーションを生成します。

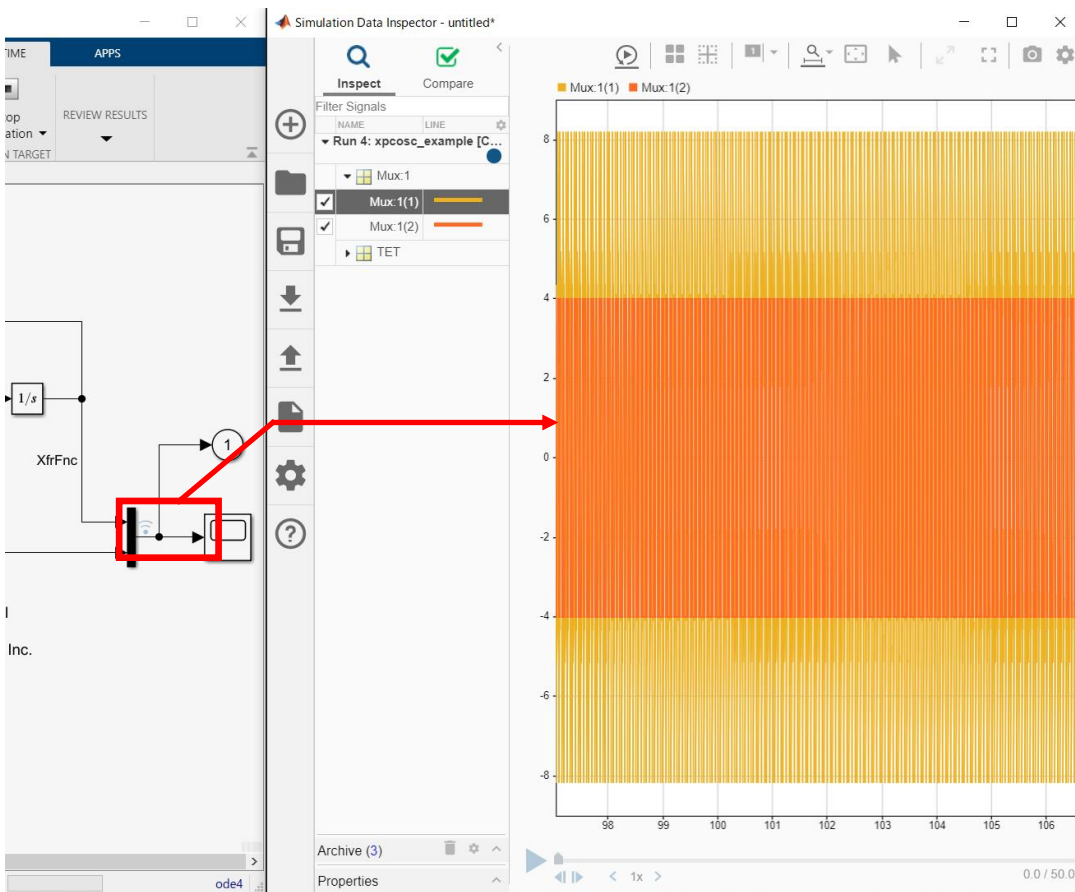
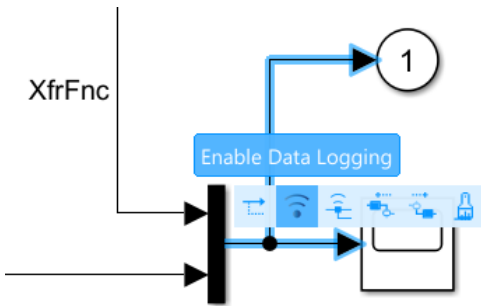
下図は R2020a の画面ですが、バージョンによって表示内容が多少異なります。



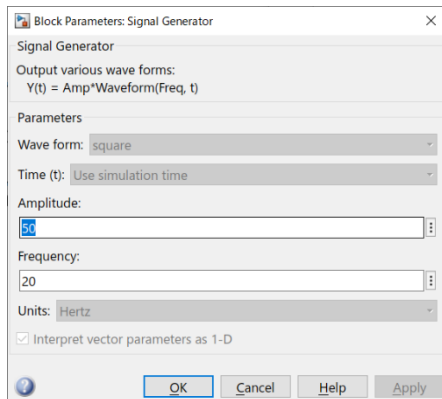
下図はビルド結果を表示する診断ビューア (Diagnostic Viewer) です。



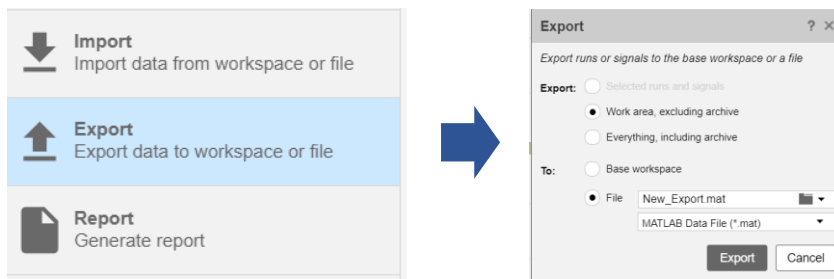
Data Inspector をクリックすると、Simulation Data Inspector が起動し、ロギング設定している信号線のデータを表示します。
 信号線を選択後に表示されるプロパティから「Enable Data Logging」を選択し、ビルドすることでその信号線をロギングすることができます。



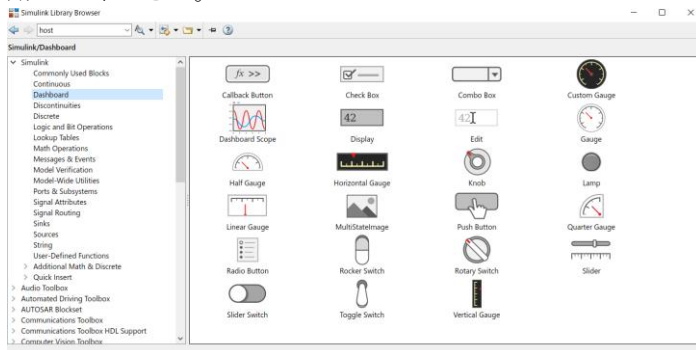
リアルタイムアプリケーションを実行中に、ブロックのパラメータを変更することで、Target computer 上のパラメータも変更されます。これにより、ロギング中のデータには変化が生じます。



Simulation Data Inspector ではロギングしたデータをファイル保存する機能もありますので、必要に応じて使用してください。



Simulink の Dashboard を使用することでも、信号やパラメータの表示や変更が可能ですので、必要に応じて使用してください。



7-3-2 サンプルモデル

Speedgoat の Target computer (Real-time target machine) を購入いただいたユーザのポータルサイトではサンプルモデルとその説明書 (Configuration and Testing Information) がダウンロードできます。

System

[Configuration and Testing Information](#)
[Simulink Test Model](#)

※ユーザごとに説明書の名称 (例: Technical Reference Manual) が異なっておりますのでご注意ください。

サンプルモデルを実行するためにはダウンロードした説明書に記載されたとおりに準備をします。

基本的には以下のような流れになります。

- ・ サンプルモデルを任意のフォルダに格納し、MATLAB 上でそのフォルダを Current Folder にした後、サンプルモデル (.slx) を開きます。
- ・ Target computer と Terminal board (端子台) をケーブルで接続します。
Terminal board とケーブルは Target computer に同梱しております。
- ・ オシレータモデルと同様にビルド、ロード、実行を行います。
- ・ Target Screen 上で正常に信号データが表示されることを確認します。



8 Control and Instrumentation

Target computer で実行しているモデル（リアルタイムアプリケーション）のブロックパラメータを設定したり、信号のデータを表示したりする方法は複数あります。

- MATLAB
- Simulink
- Simulink Real-time Explorer
- App Designer

本章では、これらの使用方法について説明します。

8-1 MATLAB

MATLAB ではコマンドウィンドウを使用して Target computer のパラメータや信号データへアクセスすることができます。

コマンドウィンドウ上で下記のように入力すると、Target computer の情報が tg に格納され、以降 tg を使用して Target computer を制御することができます。

```
>>tg=slrt
```

8-1-1 起動

下記コマンド群でモデルのビルド、Target computer にアプリケーションをロードすることや、開始・停止することを指示することができます。

```
>>rtwbuild('ModelName')
```

```
>>tg.load('ModelName')
```

```
>>tg.start
```

```
>>tg.stop
```

詳しくは Simulink Real-time のヘルプを参照してください。

8-2 Simulink

Simulink での操作方法は前章で説明しましたので、そちらをご参照ください。
これが最も簡単な操作方法となります。

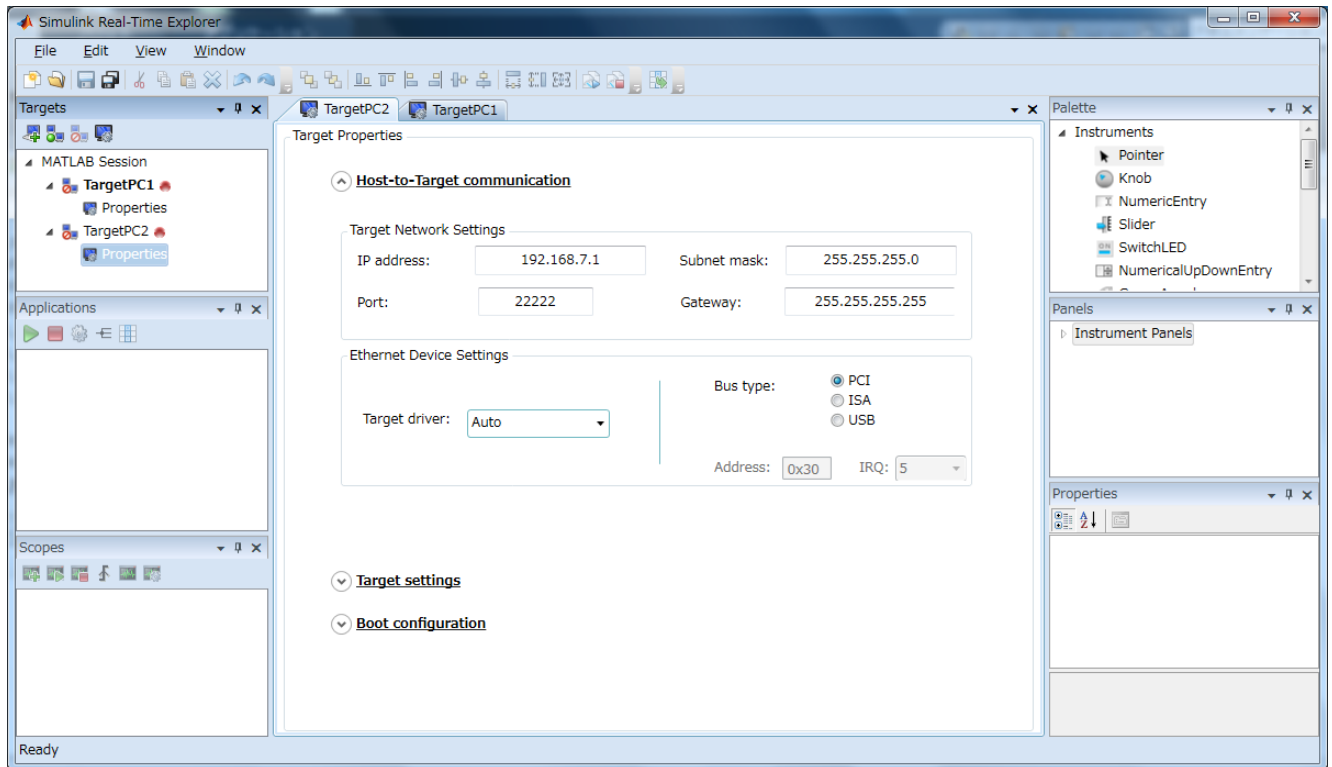
8-3 Simulink Real-time Explorer

Simulink Real-time Explorer での操作方法を説明します。

MATLAB のコマンドウィンドウで下記コマンドを実行することで Simulink Real-time Explorer が起動します。

```
Command Window  
fx >> slrtexplr
```

起動後の画面を下図に示します。

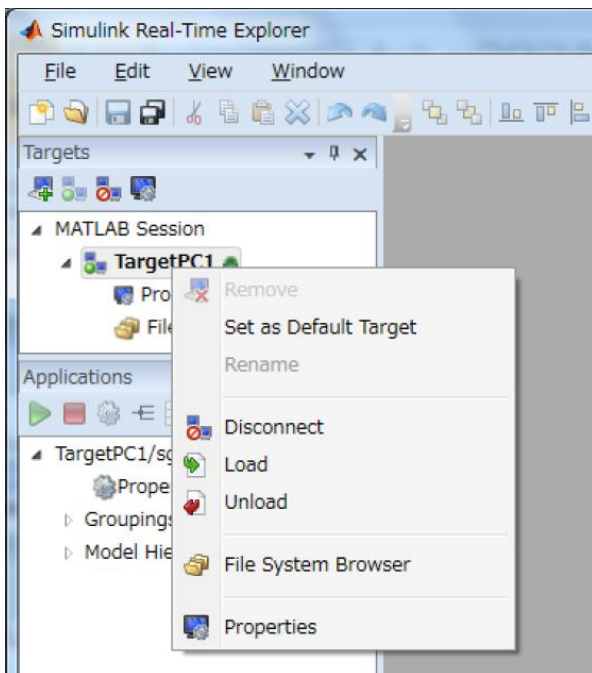


Target Computer との接続までは 7 章で説明しておりますので、そちらをご覧ください。

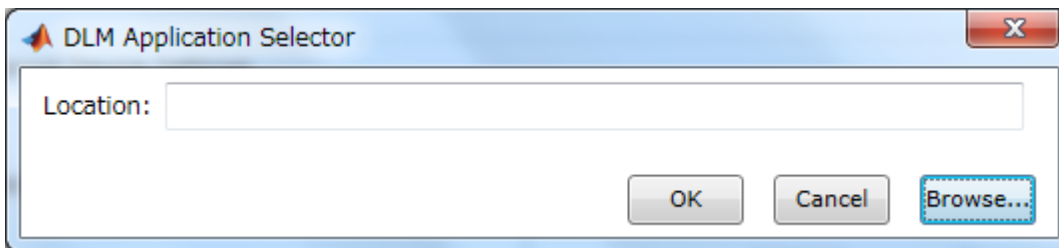
本項ではリアルタイムアプリケーションのロードから、パラメータの設定や信号の確認方法について説明します。

8-3-1 リアルタイムアプリケーションのロード

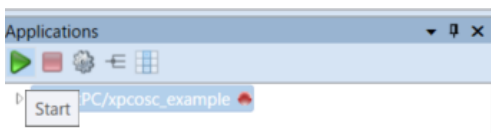
接続中の Target computer の項目を右クリックし、Load を選択します。




Browse... ボタンでロードしたいリアルタイムアプリケーションを選択し、OK ボタンをクリックします。

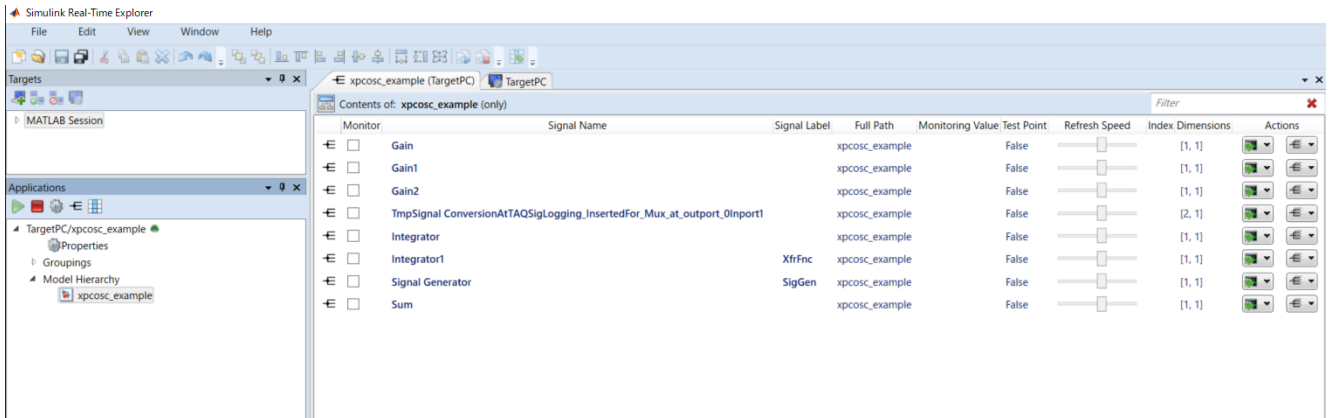


ロードが成功した後、Applications にロードしたリアルタイムアプリケーションの名称が表示されます。Start ボタンを押すことで Target computer でリアルタイムアプリケーションが実行されます。




8-3-2 信号の表示

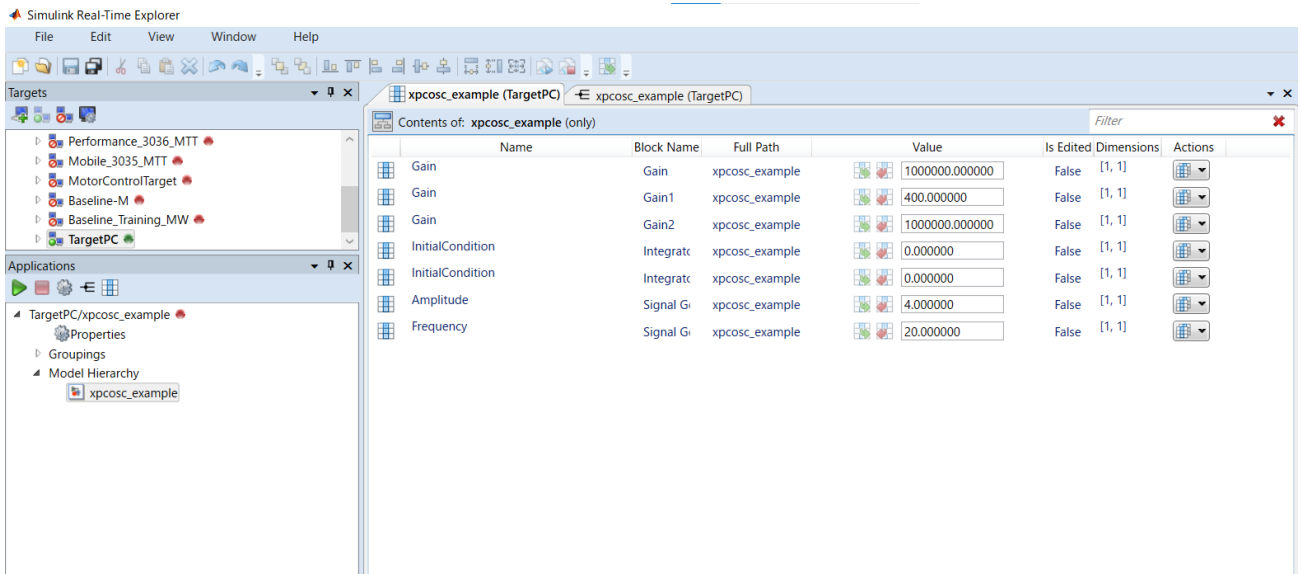
Applications の Model Hierarchy で信号を表示したいモデル階層を選択した後、View Signals ボタン  をクリックすると、下図のように Simulink モデル上の信号の一覧が表示されます。



信号を監視するためには、監視する信号の Monitor 列のチェックボックスにチェックを入れます。チェックを入れることによって Monitoring Value 列に信号のデータが表示されます。Refresh Speed を変更することで、データの更新周期を変更できます。


8-3-3 パラメータ設定


Applications の Model Hierarchy で信号を表示したいモデル階層を選択した後、View Parameters ボタン  をクリックすると、下図のように Simulink モデル上に配置したブロックパラメータのリストが表示されます。



パラメータリストから変更したいパラメータを見つけ、下方向矢印をクリックします。

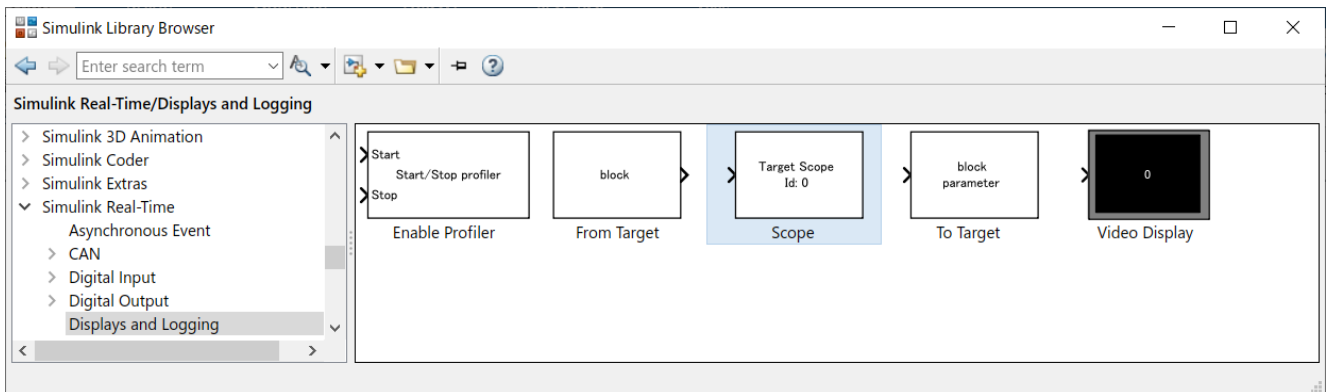
Value には、現在設定されているパラメータ値が表示されます。

パラメータを変更するには、Value に変更するパラメータ値を入力し、Apply parameter value(s) changes ボタン  をクリックします。これによりパラメータ値が変更されます。

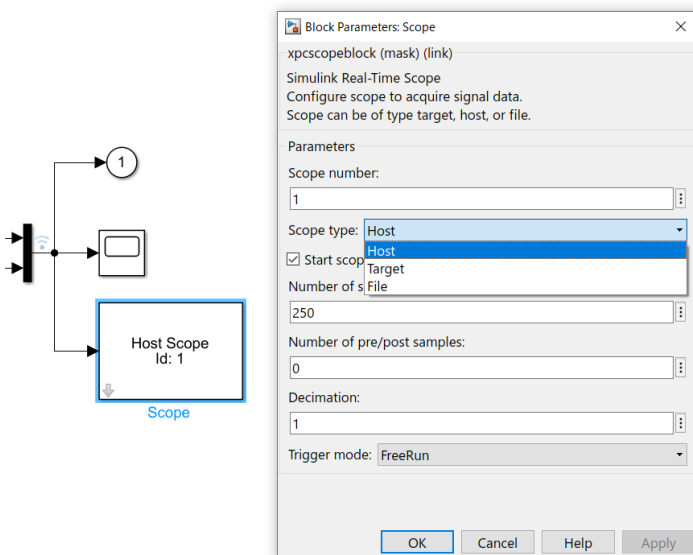
パラメータ値を以前の値に戻すには、Revert ボタン  をクリックします。

8-3-4 Host scope による信号の表示

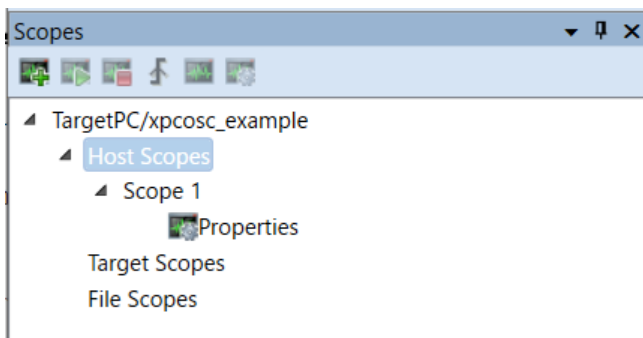
Host Scope は Simulink モデルの信号を Simulink Real-time Explorer 上にグラフ表示するために使用します。Host Scope は Simulink ブロック Scope として提供されています。




このブロックの Scope Type を Host にすることで、Host Scope として機能します。



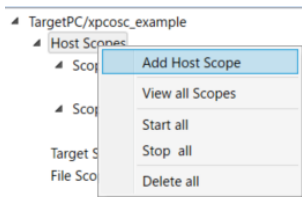
Host Scope を配置した Simulink モデルをビルドし、生成されたリアルタイムアプリケーションをロードすると、下図のように配置した Host Scope が Simulink Real-time Explorer 上に表示されます。



表示したい Host Scope にアクティブにした後、View Host Scope ボタン  をクリックすることで、グラフが表示されます。



また Add Host Scope によって、Host Scope を新規に追加することができます。



Host Scope を追加後、下記の画面で表示内容を設定します。

Id: 2 Type: HOST Status: INTERRUPTED

Sampling

Number of Samples:
 Decimation:
 Number of Pre Post Samples:

Triggering

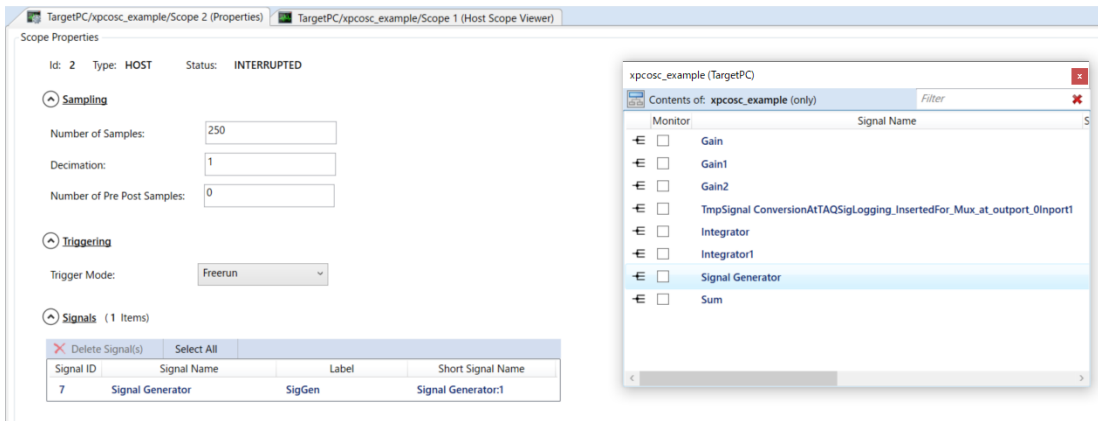
Trigger Mode:

Signals (0 Items)

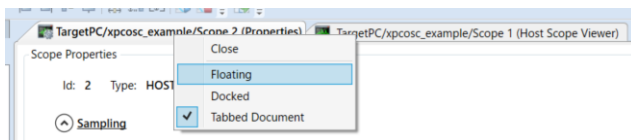
Signal ID	Signal Name	Label	Short Signal Name
<input type="checkbox"/> Delete Signal(s) <input type="checkbox"/> Select All			


Sampling と Triggering の項目は Simulink ブロックで設定する項目と同様です。項目の詳細についてはブロックのヘルプを参照してください。

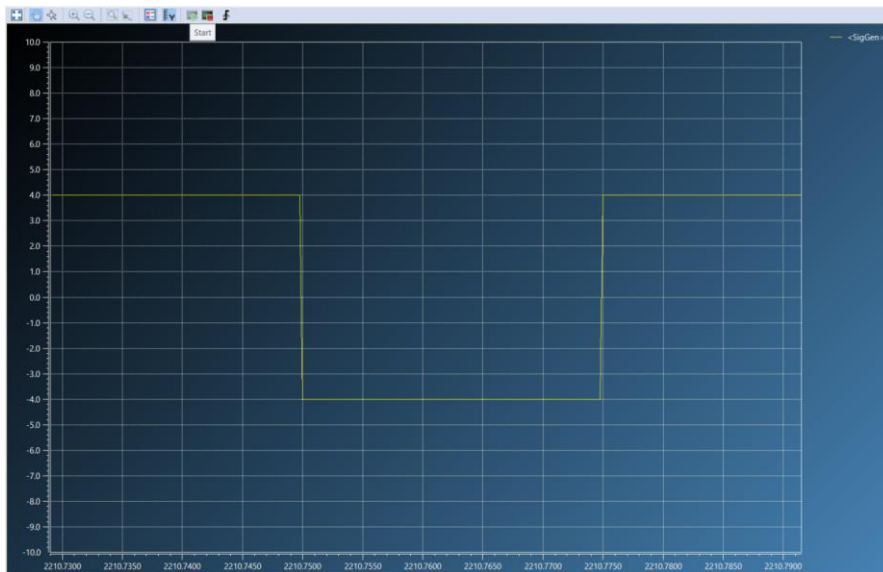
Host Scope へ表示する信号は一覧のリストからドラッグ&ドロップすることで登録することができます。



※Simulink Real-time Explorer のタブは右クリックすることでフロート表示にすることができます。



新規作成した Host Scope にアクティブにした後、View Host Scope ボタン  をクリックしてグラフを表示させます。その後 Start ボタンを押すことで信号が表示されます。

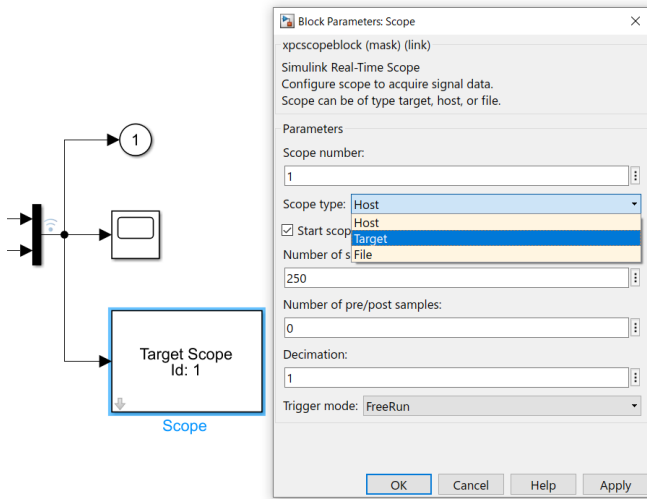


8-3-5 Target scope による信号の表示

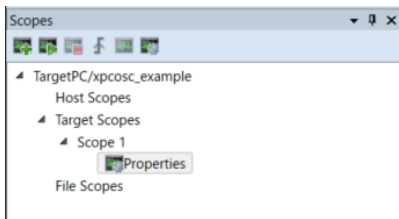
Target Scope は Simulink モデルの信号を Target Screen 上にグラフ表示するために使用します。

Target Scope は Simulink ブロック Scope として提供されております。

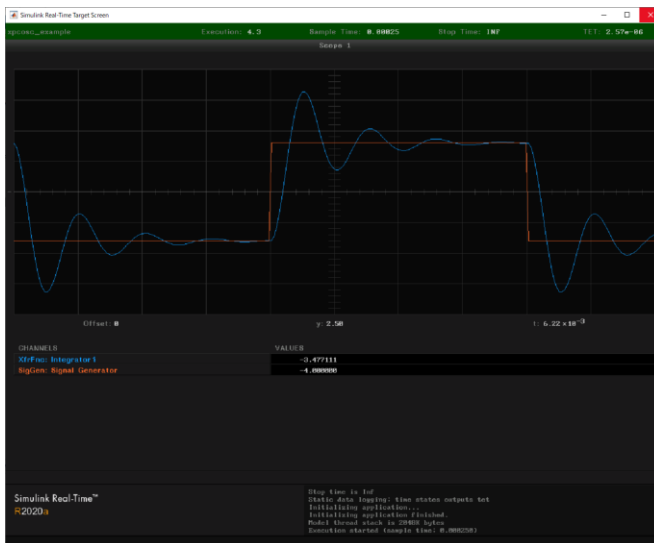
Scope Type を Target にすることで Target Scope として機能します。



Target Scope を配置した Simulink モデルをビルドし、生成されたリアルタイムアプリケーションをロードすると、下図のように配置した Target Scope が Simulink Real-time Explorer 上に表示されます。



リアルタイムアプリケーションを実行すると、Target Screen 上に Target Scope が表示されます。

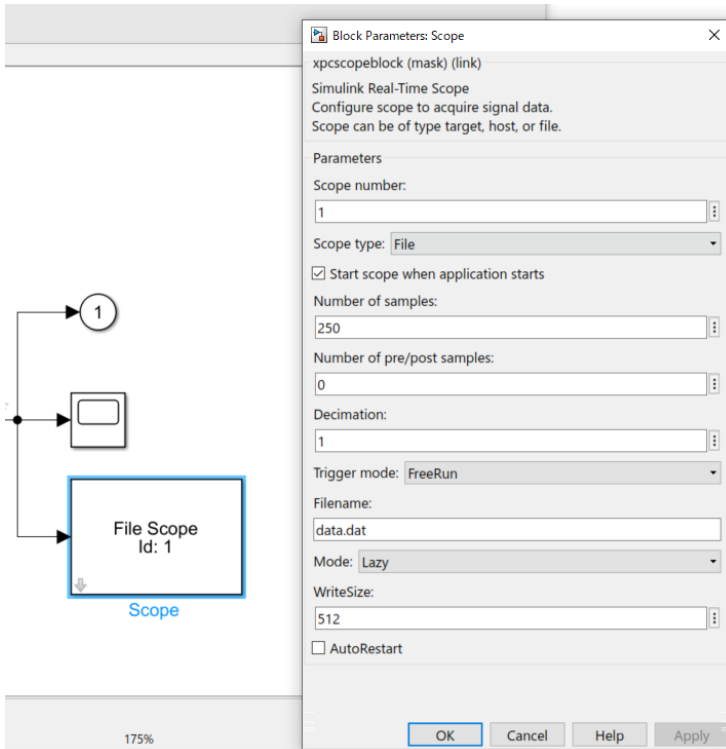


Target Scope の追加方法については Host Scope と同様です。

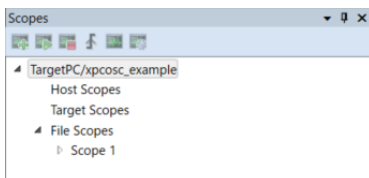
8-3-6 File scope による信号の保存

File Scope は Simulink モデルの信号を Target computer にファイル保存するために使用します。File Scope は Simulink ブロック Scope として提供されております。

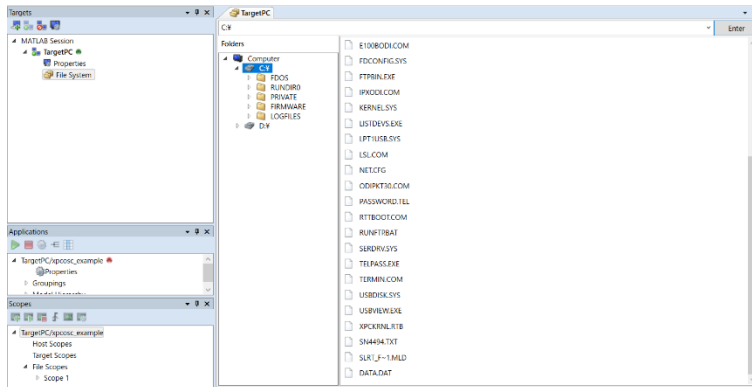
Scope Type を File にすることで File Scope として機能します。



File Scope を配置した Simulink モデルをビルドし、生成されたリアルタイムアプリケーションをロードすると、下図のように配置した File Scope が Simulink Real-time Explorer 上に表示されます。



リアルタイムアプリケーションを実行して停止すると、Target computer の C ドライブ直下にファイルが作成されます。今回は File Scope のブロック設定で Filename を data.dat と設定したので、下図で DATA.DAT が作成されていることがわかります。



Target computer と接続した状態のまま、MATLAB のコマンドウィンドウで下記コマンドを入力すると、data.dat ファイルを Host computer へ転送することができます。

```
tg=slrt;  
f=SimulinkRealTime.openFTP(tg);  
mget(f, 'data.dat');  
close(f);
```

下記コマンドを実行すると、転送した data.dat ファイルを Workspace へ展開することができます。
matlab_data = SimulinkRealTime.utils.getFileScopeData('data.dat')

8-4 App Designer

App Designer では GUI の作成と、その GUI で Target computer を制御することができます。本章では App Designer で作成した GUI アプリケーションを使用して、Target computer へモデル xpcosc をロード、実行する手順について説明します。

※本章の手順は xpcosc.slx が MATLAB の Current Folder に存在する状態で行ってください。

8-4-1 起動

MATLAB のコマンドウィンドウで下記コマンドを入力します。

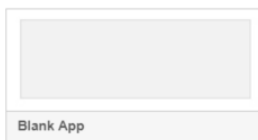
```
>>appdesigner
```

新規作成するアプリケーションの種類を選ぶことができるので、用途に合わせて選択します。今回は Blank App を選択します。

アプリケーションは xpcosc.slx と同じフォルダに作成します。

ファイル名は任意の名称 (xpcosc_app.mlapp 等) を入力します。

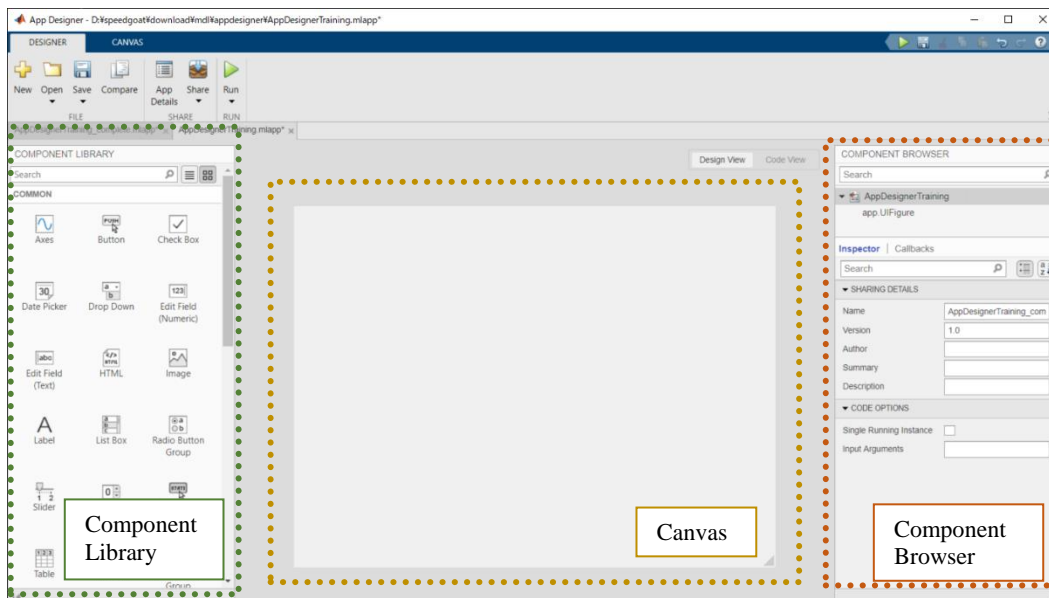
▼ New



下図が App Designer の画面となります。

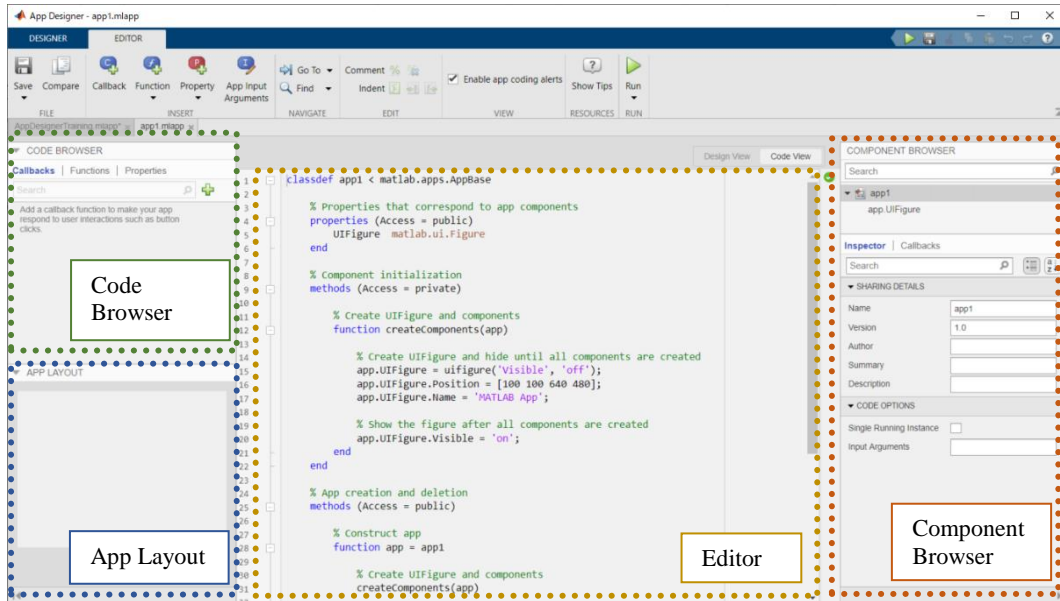
App Designer はアプリケーションの GUI の配置や設定を行う Design view とアプリケーションの挙動をコーディングする Code View という二つの画面で構成されます。

• Design view



Component Library	Canvas に配置できる GUI の部品です。
Component Browser	Canvas に配置した各 GUI の部品のプロパティやコールバックを設定します。
Canvas	GUI を配置するアプリケーションの画面です。

・ Code view



Code Browser

Editor で作成した関数などが一覧表示されます。

Component Browser

Canvas に配置した各 GUI の部品のプロパティやコールバックを設定します。

App Layout

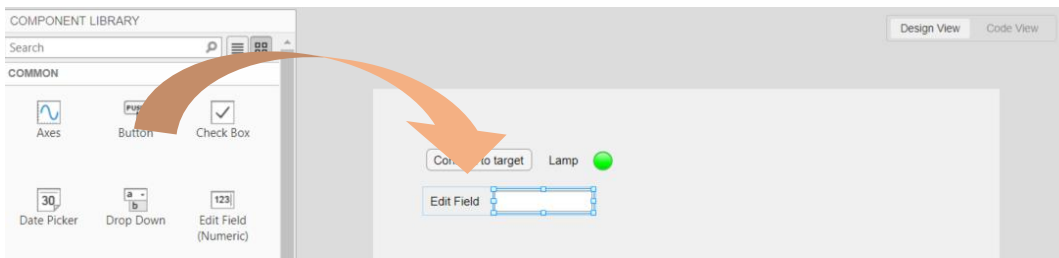
Design View の Canvas が縮小表示されます。

Editor

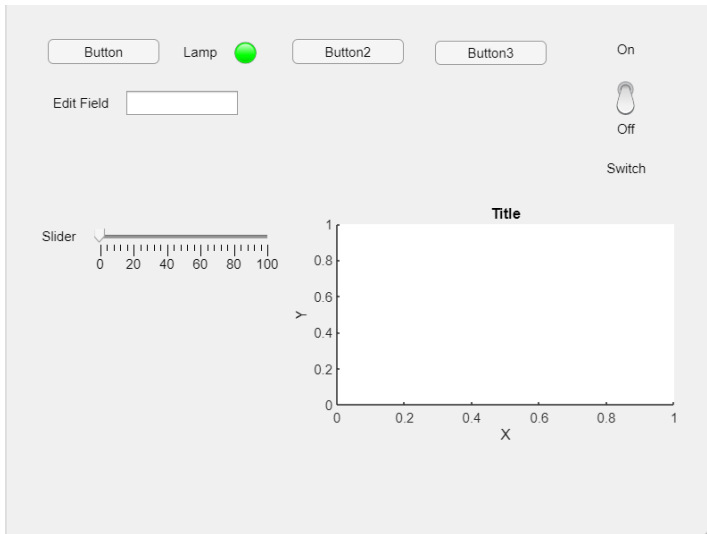
アプリケーションのコードを編集します。

8-4-2 GUIの配置

Component Library から配置したい部品を Canvas へドラッグ&ドロップします。



今回は以下のパーツを配置してください。



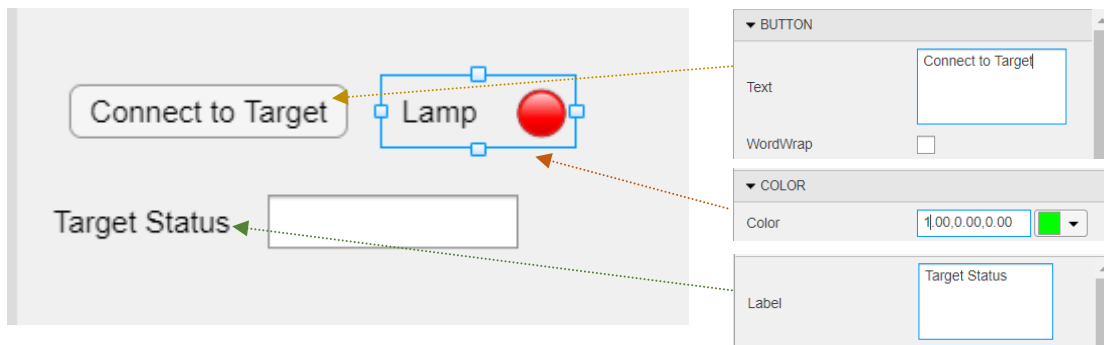
Button x3	Target computer と接続をするための Button モデル xpcosc. slx をビルドするための Button Target computer へ xpcosc のアプリケーションロードするための Button
Lamp	Target computer との接続状態を示す Lamp
Edit Field(Text)	Target computer とのあらゆる状態を示す Edit Field(Text)
Toggle Switch	Target computer へロードしたアプリケーションを実行開始/停止する Switch
Slider	Signal Generator のパラメータ Amplitude を設定するための Slider
Axes	信号 MuxOut を表示するための Axes

パーツを配置するごとに Component Browser 上でパーツに対応する変数が作成されます。

8-4-3 Target computer との接続

まずは接続に使用する Button と Edit Field の Text や Lamp の初期の色を変更します。

変更するには各パーツを選択後（アクティブにした後）、Component Browser に表示される設定を変更します。



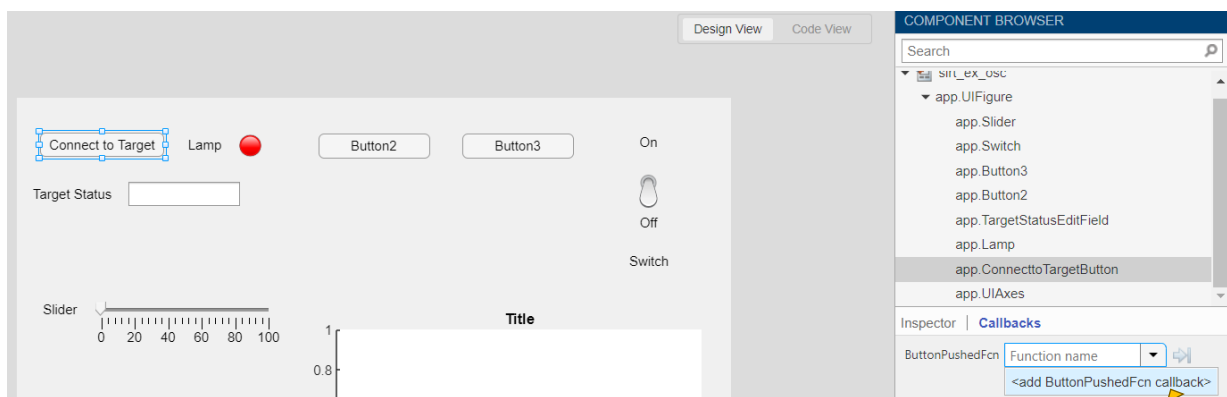
また Component Browser の設定を変更することで Code Browser 上の各パーツに対応する変数の名称も自動的に変更されます。

Connect to Target ボタンを押したときに Target computer と接続するように callback function(コールバック関数)を設定します。

Connect to Target ボタンを選択後、Component Browser の Callbacks でコールバックのイベントを選択します。

ボタンの場合は、ButtonPushedFcn というコールバックのイベントが用意されています。

これを選択すると、Code View 上でコールバック関数が作成されます。



```
% Button pushed function: ConnecttotargetButton
function ConnecttotargetButtonPushed(app, event)
```

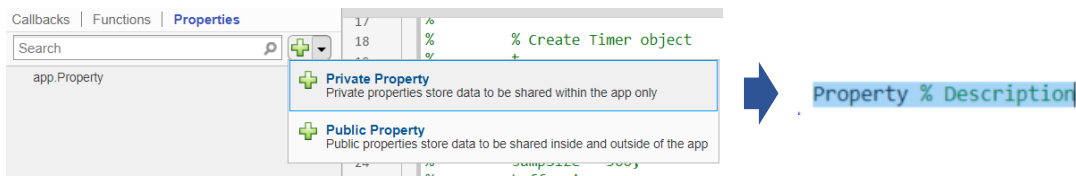
```
end
```

次にプライベート変数を作成します。

このプライベート変数に Target computer の情報を格納します。

Code Browser の Property > Private Property を選択します。

選択するとプライベート変数が作成されます。



分かりやすいように変数名を変更し、コメントを追加しておきます。

```
properties (Access = private)
    tg % Target Object
end
```

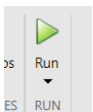
コールバック関数 ConnectToTargetButtonPushed に Target computer との接続するためのコードを追加します。

```
% Button pushed function: ConnectToTargetButton
function ConnectToTargetButtonPushed(app, event)
    app.TargetStatusEditField.Value = 'Connecting to target...';

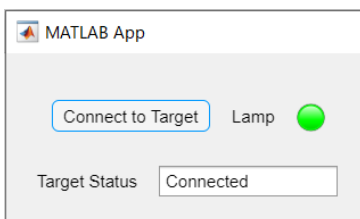
    % Get default target settings
    app.tg = slrtc;

    try
        % Connect to target
        app.tg;
        % Check if the target is correctly connected and if yes, change color of lamp to green
        if app.tg.Connected
            app.Lamp.Color = [0,1,0];
            app.TargetStatusEditField.Value = 'Connected';
        else
            app.Lamp.Color = [1,0,0];
            app.TargetStatusEditField.Value = 'Connection failed!';
        end
    catch E
        app.Lamp.Color = [1,0,0];
        app.TargetStatusEditField.Value = 'Connection failed!';
    end
end
```

コードを入力し終わったら、Run ボタンでアプリケーションを実行します。



Connect to Target ボタンを押して、正常に Target computer と接続できた場合、下記のように表示されます。



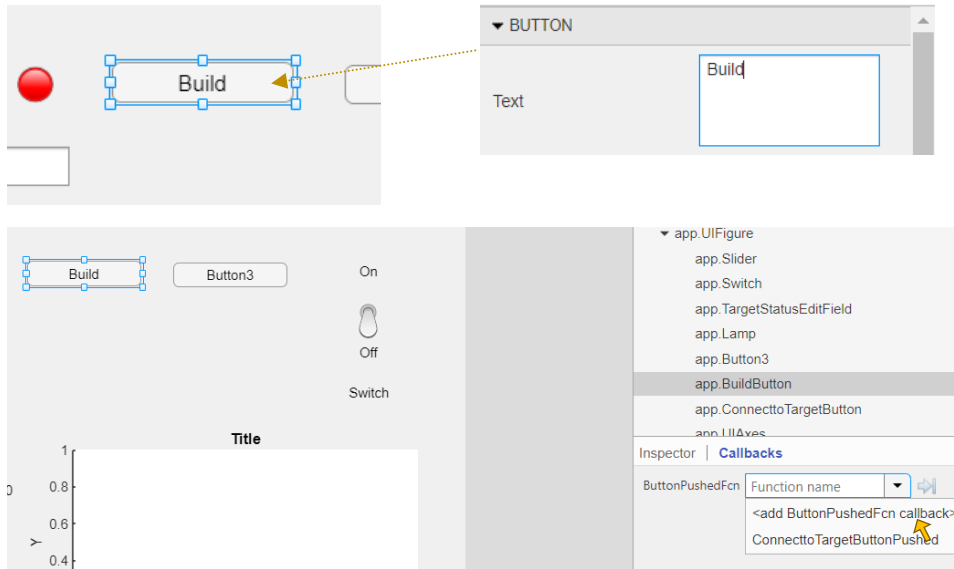
※もし接続できない場合は Simulink Real-time Explorer 上で接続できることを確認してください。

Simulink Real-time Explorer でも接続できない場合はネットワーク設定など App Designer 以外に問題があります。

8-4-4 モデルのビルド

ふたつめのボタンでモデル `xpcosc.slx` をビルドできるようにします。

Connect to Target ボタンと同様に、ふたつめのボタンの名称を変更し、コールバック関数を作成します。



またビルドするモデルの名称を格納するプライベート変数を作成します。

変数宣言時にモデルの名称も入力します。

```
properties (Access = private)
    tg % Target Object
    modelName = 'xpcosc' % Name of the Simulink model
```

※モデル名称を変更している場合は、変更後の名称を入力します。

作成されたコールバック関数 `BuildButtonPushed` にビルドを実行するためのコードを追加します。

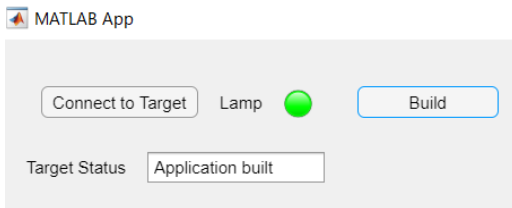
```
% Button pushed function: BuildButton
function BuildButtonPushed(app, event)
    app.TargetStatusEditField.Value = 'Building model ...'; % Update status output
    try
        rtwbuild(app.modelName);

        app.TargetStatusEditField.Value = 'Application built';
    catch E
        app.TargetStatusEditField.Value = 'Error building model.';
    end
end
```

コードの入力が完了したら `xpcosc.slx` を Simulink で開き、Configuration Parameters の Stop time を `inf` にしておいてください。これは後にモデルのパラメータの設定や信号の確認をするためです。



Run ボタンでアプリを起動して正常にビルドできた場合、下記のように表示されます。



MATLAB のコマンドウィンドウ上でもビルドの様子を確認することができます。

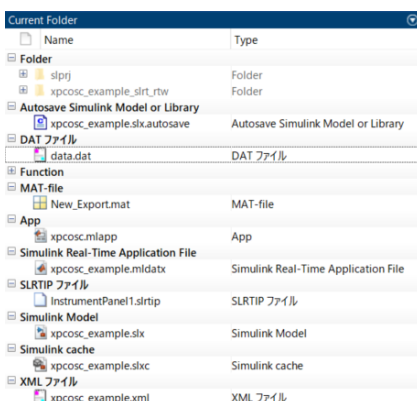
```
D:\Speedgoat\download\mdl\xpcosc_2020a\xpcosc_example_slrt_rtwinstrumented>set ~\VSCMD_START_DIR=D:\Speedgoat\download\mdl\xpcosc_2020a\xpcosc_example_slrt_rtwinstru
D:\Speedgoat\download\mdl\xpcosc_2020a\xpcosc_example_slrt_rtwinstrumented>call "C:\Program Files (x86)\Microsoft Visual Studio\2017\Professional\VC\Auxiliary\Build\VC
*****
** Visual Studio 2017 Developer Command Prompt v15.9.14
** Copyright (c) 2017 Microsoft Corporation
*****
[vcvarsall.bat] Environment initialized for: 'x86'

Microsoft(R) Program Maintenance Utility Version 14.16.27032.1
Copyright (C) Microsoft Corporation. All rights reserved.

### Linking ...
  link /NOLOGO /DLL /SUBSYSTEM:CONSOLE /DEF:xpcvcdll.def /Include:_malloc /MAP /DEBUG /IGNORE:4099 C:\PROGRAM~1\MATLAB\R2020a\toolbox\rtw\targets\cpc\target\build\
ライブラリ xpcosc_example_slrt.lib とオブジェクト xpcosc_example_slrt.exp を作成中
### Created DLL xpcosc_example_slrt.dll
C:\PROGRAM~1\MATLAB\R2020a\toolbox\slrt\target\thirdparty\OnTime\bin\mkusrdlm -c+ -+ xpcosc_example_slrt.dll ..\..\xpcosc_example
RTTarget-32 DLM Processor 6.10 (c) 1996,2017 On Time Informatik GmbH
### Created DLM ..\..\xpcosc_example.dlm

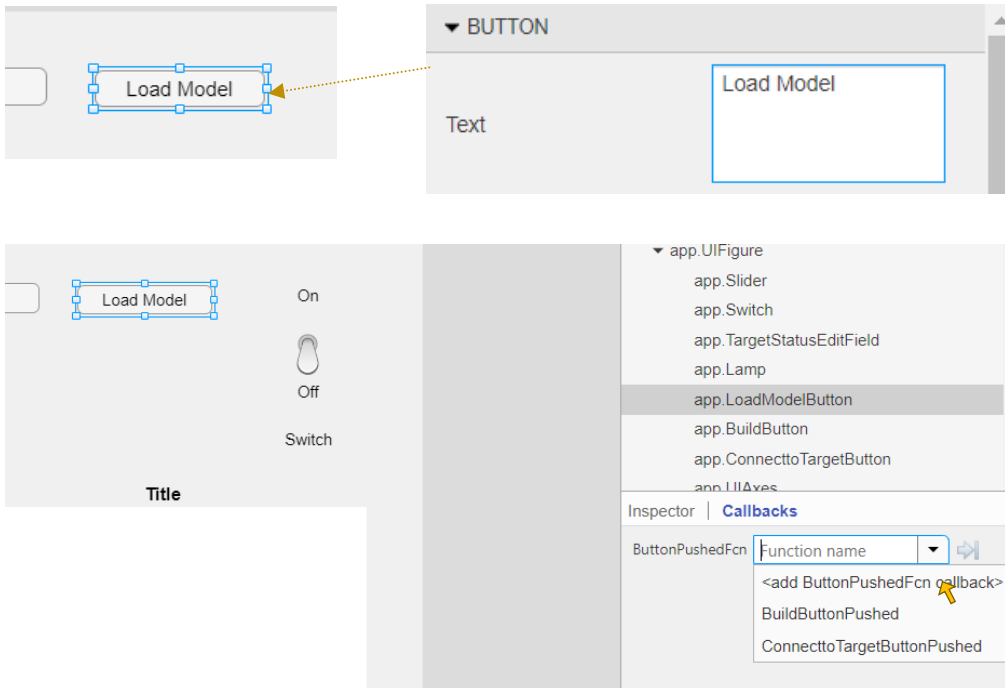
### Invoking custom build hook: CodeGenAfterMake
### Successful completion of build procedure for: xpcosc_example
### Created MLDATX ..\..\xpcosc_example.mldatx
### Invoking custom build hook: CodeGenExit
```

また下図のようにリアルタイムアプリケーション `xpcosc.mldatx` が生成されます。



8-4-5 リアルタイムアプリケーションのロード

ビルドで生成されたリアルタイムアプリケーションをロードできるようにします。
最後のボタンの名称を変更し、コールバック関数を作成します。

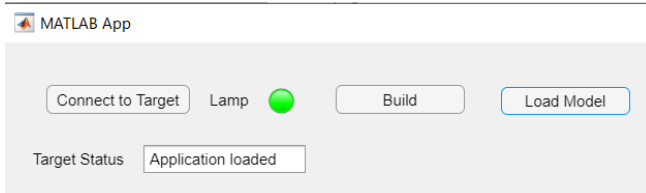


作成されたコールバック関数 LoadModelButtonPushed にリアルタイムアプリケーションをロードするためのコードを追加します。

```

% Button pushed function: LoadModelButton
function LoadModelButtonPushed(app, event)
    app.TargetStatusEditField.Value = 'Loading application...';
    try
        load(app.tg, app.modelName);
    |
        app.TargetStatusEditField.Value = 'Application loaded';
    catch E
        app.TargetStatusEditField.Value = 'Error loading application on target.';
    end
end
end
  
```

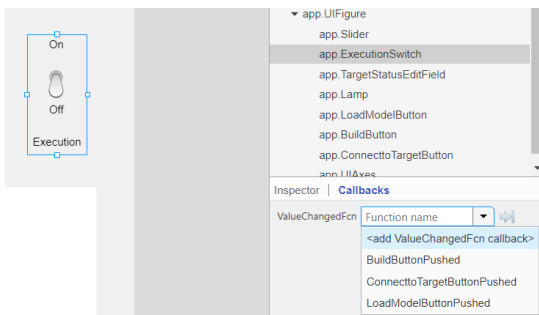
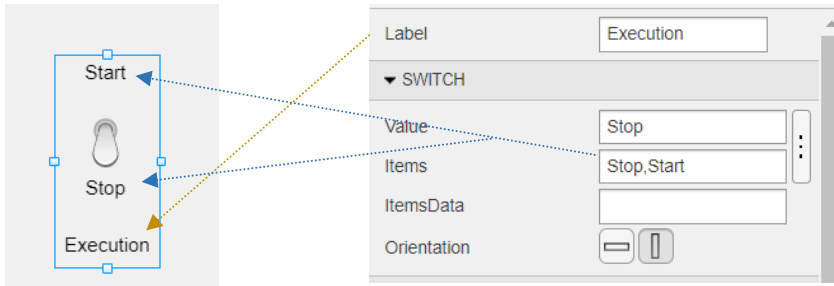
正常にロードできた場合、下記のように表示されます。



Target screen 上にもリアルタイムアプリケーションがロードされた则表示されます。

8-4-6 リアルタイムアプリケーションの実行

Target computer へロードしたリアルタイムアプリケーションを開始・停止できるようにします。
トグルスイッチの名称を変更し、コールバック関数を作成します。



作成されたコールバック関数 ExecutionSwitchValueChanged にリアルタイムアプリケーションを開始・停止するためのコードを追加します。

```

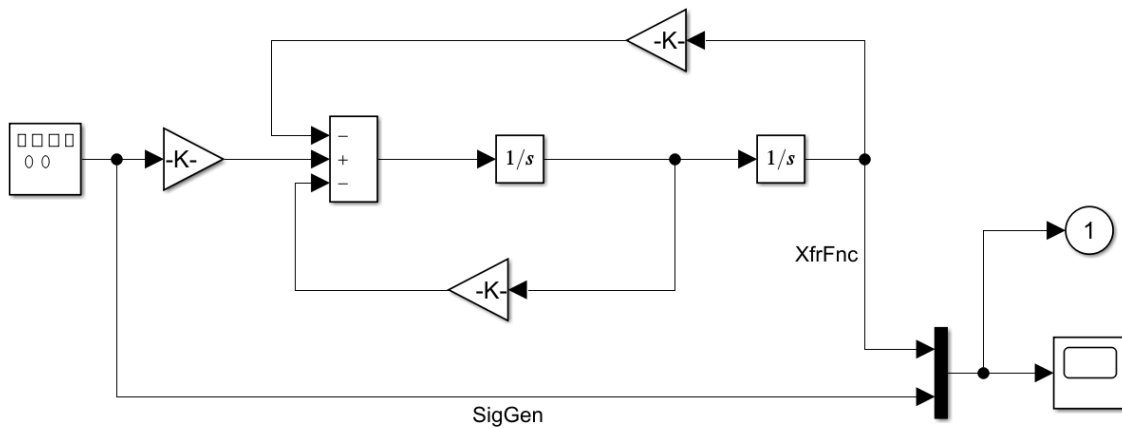
% Value changed function: ExecutionSwitch
function ExecutionSwitchValueChanged(app, event)
    value = app.ExecutionSwitch.Value;
    switch value
        case 'Start'
            start(app.tg);      % Start real-time application
            app.TargetStatusEditField.Value = 'Application running';
        case 'Stop'
            stop(app.tg);      % Stop real-time application
            app.TargetStatusEditField.Value = 'Application stopped';
    end
end
    
```

正常に開始できた場合、Target screen 上の Execution にリアルタイムアプリケーションを実行している時間が表示されます。

8-4-7 信号の表示

Simulink モデルのブロック線図上の信号線の値を表示できるようにします。

今回は下図の Signal Generator と Integrator1 ブロックの出力 (SigGen と XfrFnc) を表示します。

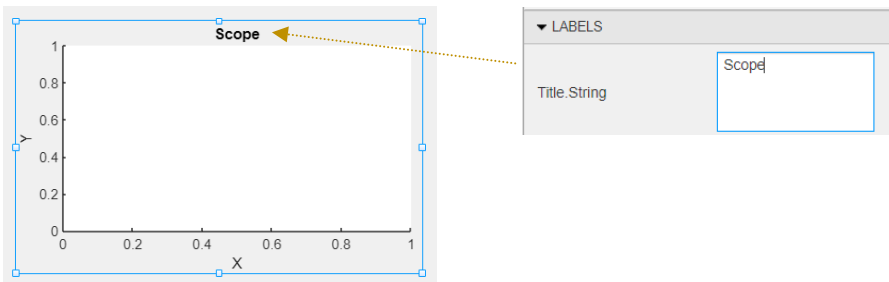


Model xpcosc
Simulink Real-Time example model

Copyright 1999-2013 The MathWorks, Inc.

信号ラインの値は作成しておいた Axes に表示します。

Axes のタイトル名称を変更しておきます。



次に Axes に表示するデータが格納される Host Scope を作成します。

```
properties (Access = private)
    tg % Target Object
    modelName = 'xpcosc_example' % Name of the Simulink model

    scp_01; % Host scope for data display
    sampSize=1000; % Scope frame size
```

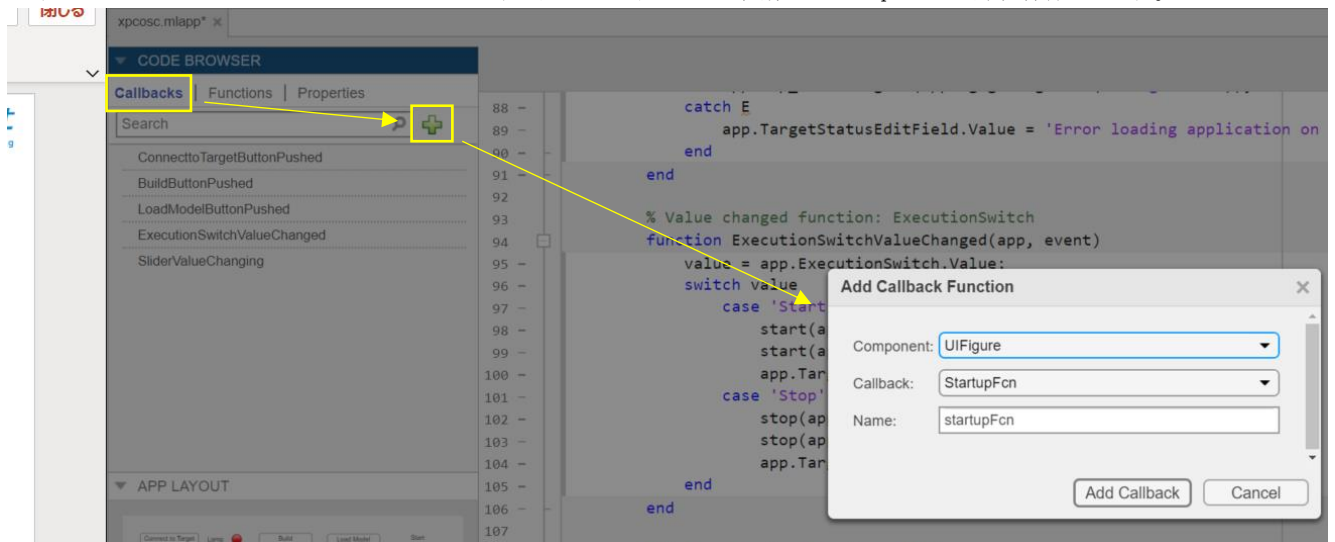
リアルタイムアプリケーションのロード時に、Host Scope の設定をまとめて行うコードを追加します。

```
% Button pushed function: LoadModelButton
function LoadModelButtonPushed(app, event)
    app.TargetStatusEditField.Value = 'Loading application...';
    try
        load(app.tg, app.modelName);

        app.TargetStatusEditField.Value = 'Application loaded';

        % Add a host scope and add signals to log
        app.scp_01 = addscope(app.tg, 'host');
        app.scp_01.Decimation = 1;
        app.scp_01.TriggerMode = 'FreeRun';
        app.scp_01.NumSamples = app.sampSize;
        app.scp_01.addsignal(app.tg.getsignalid('Signal Generator'));
        app.scp_01.addsignal(app.tg.getsignalid('Integrator1'));
    catch E
        app.TargetStatusEditField.Value = 'Error loading application on target.';
    end
end
```

次に Host Scope で取得した信号線のデータを周期的に Axes で表示するためのタイマーを追加します。まずは Code Browser でアプリケーション開始時に実行される関数 StartupFcn を新規作成します。

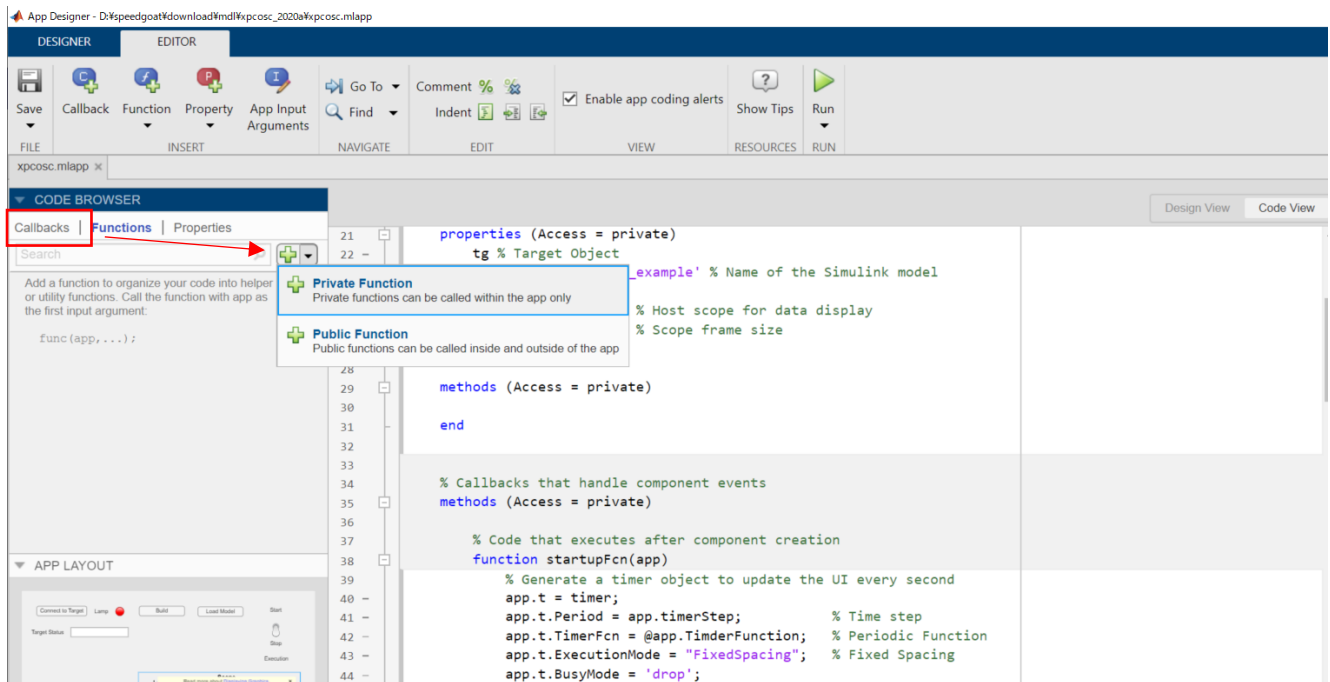


StartupFcn にタイマーを作成するコードを追加します。追加したコードにより周期的に関数 TimerFunction を実行します。

```
% Code that executes after component creation
function startupFcn(app)
    % Generate a timer object to update the UI every second
    app.t = timer;
    app.t.Period = app.timerStep;           % Time step
    app.t.TimerFcn = @app.TimerFunction;    % Periodic Function
    app.t.ExecutionMode = "FixedSpacing";   % Fixed Spacing
    app.t.BusyMode = 'drop';
end
```

関数 TimerFunction を新規作成します。

Cord Browser で Private Function を新規作成します。



新規作成した Private Function の名称を TimerFunction へ変更します。

```
function results = TimerFunction(app,~,~)
end
```

TimerFunction にコードを追加します。

このコードにより、Host Scope から信号線のデータや経過時間を取得し、それを Axes に反映させます。

```
function results = TimerFunction(app,~,~)
% Get data from host scope and plot in the graph
if strcmp(app.scp_01.Status, 'Finished')
% Update the plot buffer with the new host scope data chunk
app.plotTime = app.scp_01.Time;
app.plotData = app.scp_01.Data;
% Restart the scope
app.scp_01.start;
% Update time axis limits
app.UIAxes.XLim = [app.plotTime(1), app.plotTime(end)];
% Plot the new buffered data
plot(app.UIAxes, app.plotTime, app.plotData);
legend(app.UIAxes, 'SigGen', 'XfrFnc', 'Location', 'Northeast');
end
end
```

最後に Host Scope と Timer を開始するコードを追加します。

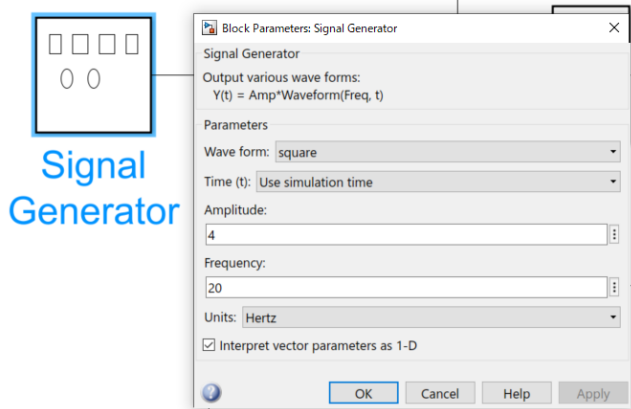
```
% Value changed function: ExecutionSwitch
function ExecutionSwitchValueChanged(app, event)
value = app.ExecutionSwitch.Value;
switch value
case 'Start'
start(app.tg); % Start real-time application
start(app.t); % Start timer
start(app.scp_01); % Start scope
app.TargetStatusEditField.Value = 'Application running';
case 'Stop'
stop(app.tg); % Stop real-time application
stop(app.t); % Stop timer
stop(app.scp_01); % Stop scope
app.TargetStatusEditField.Value = 'Application stopped';
end
end
```

アプリケーションを実行して、下図のように表示されれば正常に動作しています。

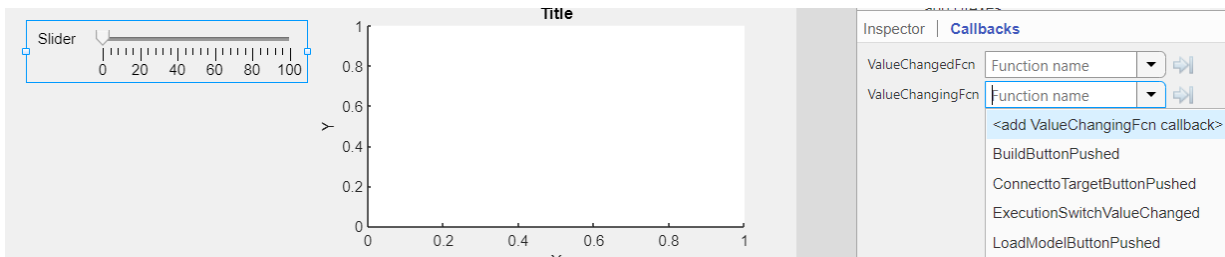


8-4-8 パラメータの設定

Simulink モデルのブロックのパラメータを変更できるようにします。
今回は下図の Signal Generator の Amplitude を変更対象とします。



まずは Slider のコールバック関数を追加します。
Slider にはふたつのイベントにコールバック関数を設定できます。
今回は ValueChangingFcn を追加します。



Amplitude を設定するためのコードを追加します。

```
% Value changing function: Slider
function SliderValueChanging(app, event)
    changingValue = event.Value;
    app.tg.setparam('Signal Generator', 'Amplitude', changingValue);
end
```

アプリケーションを実行し、スライダを操作してください。
スライダの値に応じて、Axes の縦軸の幅が変動すれば正常に動作しています。

