SWITCHERGEAR
GEAR TO CONTROL YOUR POWER SWITCHES

Graphical Coding With Scilab

How to install and use Scilab and X2C with SwitcherGear

APPLICATION NOTE

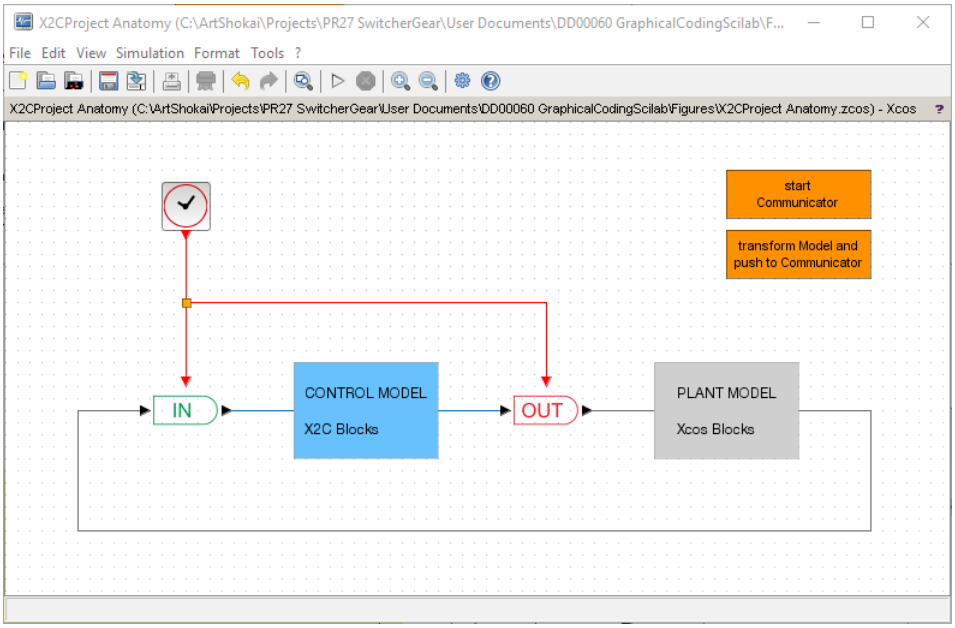
Table of Contents

contact@denki.com.au

Introduction

This document shows how to develop control code for the SwitcherGear controller using the Scilab/Xcos graphical modelling environment and the third-party X2C tools.

The following figure shows the general structure of a X2C project. The graphical model directly resembles a control system block diagram, where the output of the controller drives the plant, and the state of the plant is measured and fed back as the input to the controller.



The X2C project can be used in two ways:

1. **Real-Time Target** The control model is converted to C code and is executed in real-time on the SwitcherGear controller hardware. The *real-time* control model controls a *physical* plant (e.g. a motor, a 3-phase grid, etc.) and the plant model is ignored.

   See Sections Execution on a Real-Time Target and Real-Time Interaction for details.

2. **Off-line Simulation** The control model and the plant model are both executed in the Scilab/Xcos simulation environment. The *simulated* control model controls the *simulated* plant model.

   See Section Off-line Simulation for details.

The first sections of this document provide information about:

- Installing the required software.
- Preparing the SwitcherGear controller.
- Using the software tools Code Composer Studio and Scilab/Xcos.
- Building a control model in X2C.

Installing Software

Scilab

Scilab is a free and open source software for numerical computation that provides a powerful computing environment for engineering and scientific applications. Scilab includes Xcos, which is a graphical editor to design hybrid dynamical systems models that can be loaded, saved, compiled and simulated. Scllab/Xcos provides features similar to other commercial numerical computation packages.

IMPORTANT	You must install Scilab version 5.5.2. This is not the latest release version, but is the version required to support the X2C package.

Download the installer from

https://www.Scilab.org/en/download/previous

There are installers for Windows 32-bit and 64-bit.

When using the menu bar in Scilab, be aware that the available menus change depending on which pane in the Scilab user interface is active. Unless otherwise stated, you should ensure that the Scilab Console pane is active to access menus described in this documentation.

MinGW Compiler

To run your model as a simulation in the Scilab environment, the model must be compiled to run on your PC's processor. For this purpose, you must install the MinGW compiler, which is the preferred compiler for Scilab.

Download the compiler that matches your Windows and Scilab installation, and install.

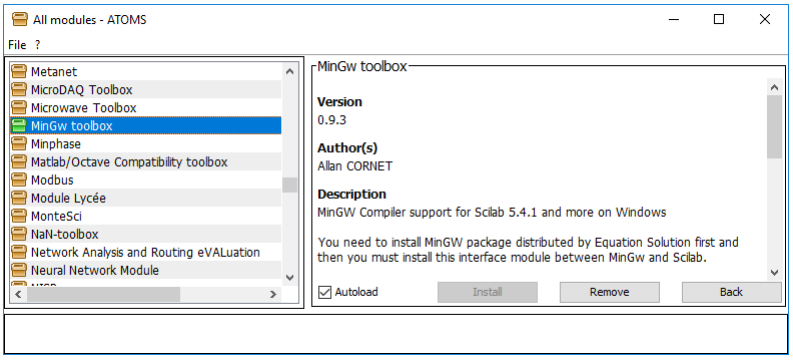
Platform	Scilab	Installer file
Windows 32-bit	Scilab 32-bit	http://atoms.Scilab.org/toolboxes/mingw/0.9.3/files/gcc-4.6.3-32.exe
Windows 64-bit	Scilab 32-bit	
Windows 64-bit	Scilab 64-bit	http://atoms.Scilab.org/toolboxes/mingw/0.9.3/files/gcc-4.6.3-64.exe

When installing, you should choose to install into the default path (**`/gcc`**).

MinGW Toolbox for Scilab

The MinGW Toolbox enables interaction between Scilab and the MinGW Compiler. The toolbox is installed using the ATOMS module manager in Scilab.

3. Click on the Scilab 5.5.2 Console pane – this displays the correct menu for the next step.
4. Select the menu item **Applications > Module manager - ATOMS**, to open the Module Manager dialogue box.
5. In the navigation panel on the left side, click on the All Modules folder item to show all available modules.
6. Scroll down the list and click on the MinGw Toolbox item to show the following:



- Click on the **Install** button.

X2C

X2C is a tool developed by the Linz Centre Of Mechatronics for the model-based development and code generation of real time control algorithms for microcontrollers, including Texas Instruments C2000. X2C uses the open source environment Scilab/Xcos to build the graphical control model. With Scilab/Xcos, the model can be simulated and the behaviour of the algorithm can be validated before it is executed on the target microcontroller.

Libraries with numerous blocks are available for creating the graphical model. C-code based on the model is generated by X2C and then compiled with the C2000 code generation tools. X2C allows easy model tuning and debugging with the included applications Communicator and Scope. With these tools, model parameters and data can be updated and monitored in real-time.

The X2C library is available as a free download, which contains the standard features required to build control models for SwitcherGear controllers.. There are also paid versions that include more advanced modelling blocks.

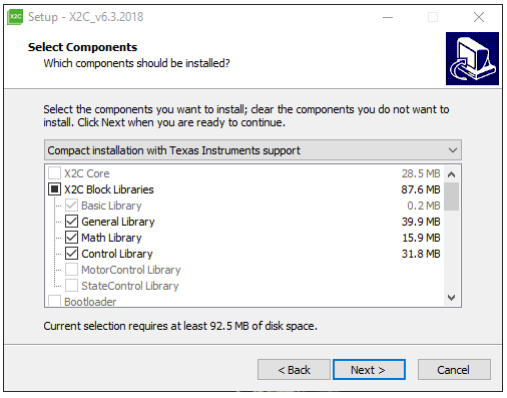
Download the latest version from

http://www.mechatronic-simulation.org/

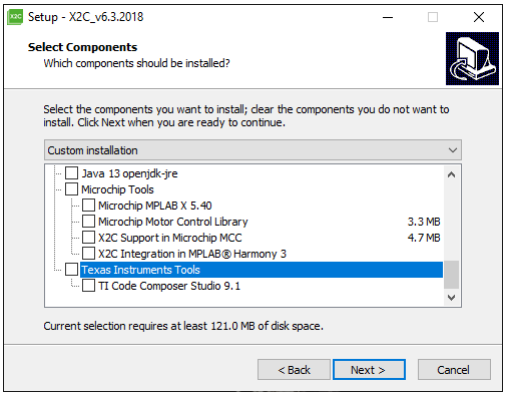
X2C version 6.3 (free) was used to capture images for this document.

Installation

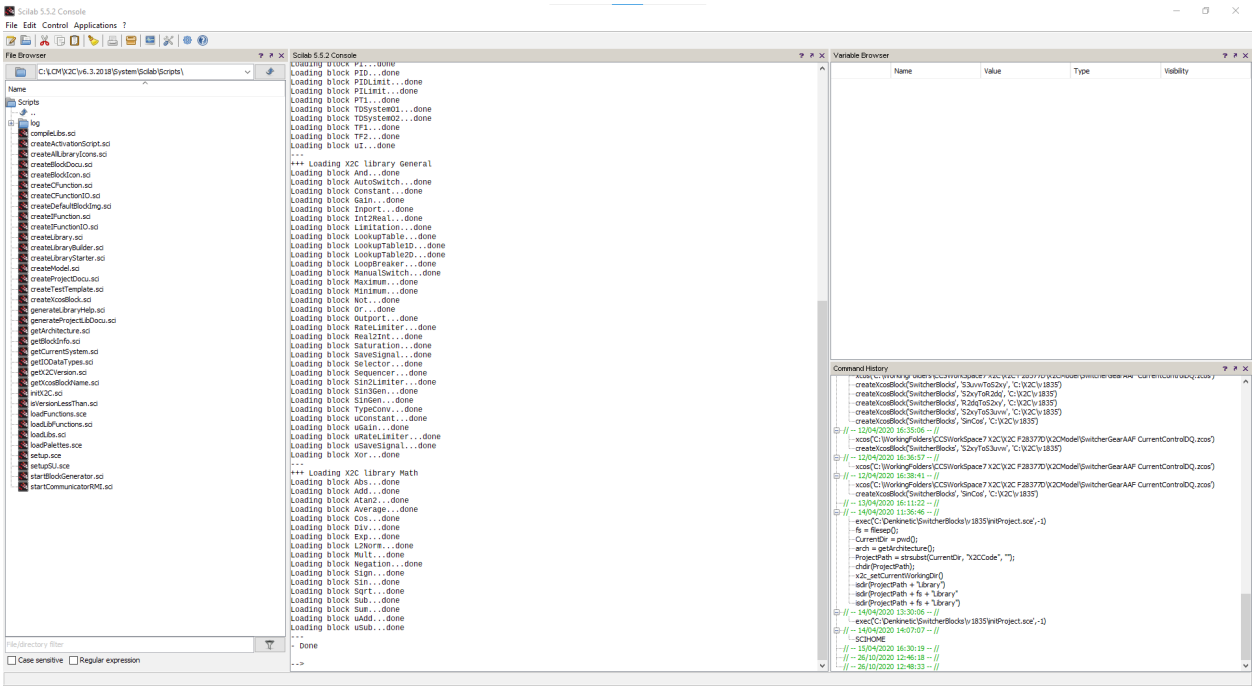
1. Run the installer application.
2. In the **Select Destination Location** pane, use the default install location.
3. In the **Select Components** pane
   a. Choose from drop-down list “Compact installation with Texas Instruments support”.



   b. Scroll to the bottom of the checkbox list and deselect “Texas Instruments Tools”. (These tools will be installed in the next section of this document.)



4. Continue to the finish of the installation
   a. Allow the installer to run Scilab.
5. Scilab will open and execute the X2C installation scripts.
6. To complete the installation, restart Scilab (close and re-open).
   a. The Scilab **Console** pane will show the loading of X2C blocks, and then print Done.



Documentation

Scilab tutorials https://www.Scilab.org/tutorials?field_tutorials_tid=15

Code Composer Studio

To run your model as a real-time controller on a SwitcherGear hardware target, the model must be compiled to run on a Texas Instruments C2000 industrial microcontroller. For this purpose, you must install the Code Composer Studio (CCS) software from Texas Instruments, which includes the build tools for C2000.

Download the latest version from

http://www.ti.com/tool/CCSTUDIO

When installing, you should choose to install into the default path (**`/ti`**). You should customise the installation by selecting only the tools for the C2000 family of microcontrollers.

Documentation

Code Composer Studio User's Guide

A collection of links to other user guides and documentation resources Code Composer Studio Documentation Overview

C2000Ware

The C2000Ware software from Texas Instruments includes the device support files required to build C2000 applications.

Download the installer from

http://www.ti.com/tool/C2000WARE

When installing, you should choose to install into the default path (**`/ti/c2000`**).

SwitcherWare

The SwitcherWare Library provides C/C++ code resources that simplify the creation of industrial grade applications for power converter systems. The library includes wrapper classes that provides developers with high-level access to the powerful features of the CPU core and hardware peripherals of the Texas Instruments C2000 microcontrollers. The library also provides resources to simplify the use of SwitcherGear controller and hardware modules.

A licence to use the SwitcherWare Library is attached to each SwitcherGear controller. The SwitcherWare Library can be used to build applications for generic C2000-based controllers – please contact Denkinetic for licensing.

The SwitcherWare Library is available from Denkinetic.

When installing, you should choose to install into the default path (**`/Denkinetic/SwitcherWare`**).

Documentation

Documentation for the SwitcherWare Library is included in the install package and is located at

`<LibrarySwitcherWare>/SwitcherWare_<version>/document`

SwitcherBlocksX2C

The SwitcherBlocksX2C Library provides additional blocks for use with X2C and example projects for SwitcherGear controllers.

The SwitcherBlocksX2C Library is available from Denkinetic.

When installing, you should choose to install into the default path (**`/Denkinetic/SwitcherBlocksX2C`**).

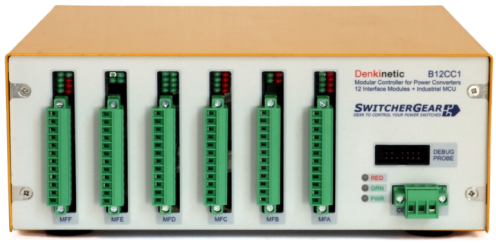
SwitcherGear Real-Time Target

Power Converter System

Your power converter system includes the power devices, sensors, energy sources, loads, etc. that are assembled according to the topology required for your application. The digital control algorithm for this physical system will be implemented as a graphical model (using X2C) and targeted to run on the SwitcherGear controller hardware.

SwitcherGear Controller

The SwitcherGear controller contains a C2000 microcontroller that runs the real-time controller model, and hardware interface modules that connect the microcontroller to the external power converter system. Typical module interfaces include PWM outputs for power converters, sensor inputs for voltage and current measurements, encoder inputs for motor shaft position measurement, etc.



SwitcherGear Configuration

Each SwitcherGear controller is configured with particular modules and has a particular routing of internal signals between the modules and the MCU. These are documented in the SwitcherGear Configuration Document for your controller. The three letter configuration code on the under-side of your controller can be used to identify the correct document.

You should use this document to understand the configuration of your SwitcherGear controller. This will make it much simpler to make the external wired connections to the converters, sensors, etc. of your power system. It will also provide insight into the software you develop for your control model.

Refer to Application Note SwitcherGear Configuration and Signal Chains (DD00061).

Debug Probe

A debug probe is required to program the microcontroller in the SwitcherGear controller. XDS110 class debug probes are low cost devices with good performance, and are recommended for use with the SwitcherGear controller. They can be included in a bundle with the SwitcherGear controller, or purchased separately from various vendors.

Connect the ribbon cable to the Debug Probe port on the front of the SwitcherGear controller. For the XDS110 debug probe, you must use the supplied 14-pin adaptor.

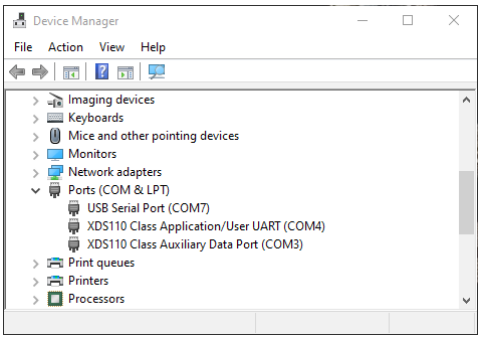
Serial Port

Your configuration must include a RS-232 interface module if you wish to use the interactive tools from X2C. The X2C Communicator tool allows model parameters (e.g. controller gains, reference values, etc.) to be changed on-the-fly. The X2C Scope virtual oscilloscope tool allows digital controller variables to be captured in real-time, visualised on the PC and saved for further numerical processing.

A USB-to-Serial adaptor is the simplest way to add a serial port to a PC or laptop.

Wire the serial cable to the N23201 module on the SwitcherGear controller. Refer to the Reference Manual for wiring information.

You must determine the port number that the serial port has been allocated by the Windows operating system. In Windows, use the Device Manager to find the port number, which will be listed under **Ports (COM & LPT)**. The port number is in the format “COMX”, where X is a unique number.



The port settings (i.e. Baud rate, etc.) will be managed by the X2C Communicator application, so do not need to be set here.

Current and Voltage Sensors

AIN004 sensor interface modules are used to interface current and voltage sensors to the SwitcherGear controller. The module has configurable settings for the measurement range and mode. You must ensure that the settings match the chosen sensors and the application. Please refer to the Module AIN004 Reference Manual for details on how to choose the settings.

For each sensor channel of the AIN004 module you must:

1. Determine the maximum range of the current or voltage that is to be measured.
2. Choose an appropriate sensor to measure the current or voltage.
3. Determine the corresponding range of the output current signal of the sensor.
4. Select a suitable input range and input mode for the sensor channel.
5. Determine the correct settings for the sensor channel on the AIN004 module.
6. Ensure that the sensor channel is correctly configured with the settings.
7. Calculate the measurement range of the sensor.
8. In the **`Signals.cpp`** file, identify the **`AinPinScaled`** object constructor that corresponds to the sensor channel.
9. Ensure that the measurement range of the sensor is set as the arguments of the object.

For example, current sensor is required to measure a line current up to 15 A RMS. The sensor will be connected to sensor channel 0 of an AIN004 module that is installed in module slot MRB.

1. For a 15 A RMS current, the maximum range of the current is –21.2 A to +21.2 A.
2. A suitable sensor is the SNI005 current sensor from Denkinetic. The sensor can measure AC and DC currents, and has an input capability of 25 A RMS and 50 A peak. The output current range of the sensor is ±50 mA for an input current range of ±50 A.
3. Based on the sensor data, the range of the output current signal of the sensor is -21.2 mA to + 21.2 mA.
4. The sensor channel should be configured for a 30 mA input range and bipolar input mode.
5. (Refer to the Module AIN004 Reference Manual for settings.)
6. Consult the SwitcherGear Configuration Document for your controller, or open the controller to check the jumper settings. Modify the jumpers if required.
7. Based on the sensor channel settings and the characteristics of the sensor, the measurement range of the sensor is -30 A to +30 A.
8. In the AAF_**`Signals.cpp`** file, the **`AinPinScaled`** object constructor that corresponds to this sensor channel might look like this:

```
AinPinScaled MRB_sensorA0(ADCINC3, -0.02, 0.02);
```

9. Modify the arguments to match the measurement range:

```
AinPinScaled MRB_sensorA0(ADCINC3, -30.0, 30.0);
```

Documentation

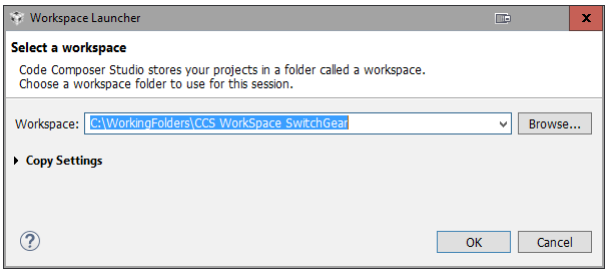
Reference manuals for hardware and application notes can be downloaded from

http://denki.com.au/download.html

Using Code Composer Studio

Workspaces

When you open CCS for the first time, you will be asked to provide a path for a Workspace. A workspace is used to group together related projects and code resources.



You can have as many workspaces as you require, to keep

Build Variables

Build variables are used in a CCS workspace to store project-related data, such as the install paths of code libraries.

The easiest way to set build variables is to import them from a vars.ini file. Suitable vars.ini files are installed with firmware libraries supplied by Denkinetic. The vars.ini file is located in the top-level install folder for the library.

1. For each library, edit the vars.ini file and update the following lines to match the actual location and version number of the library installed on your system.

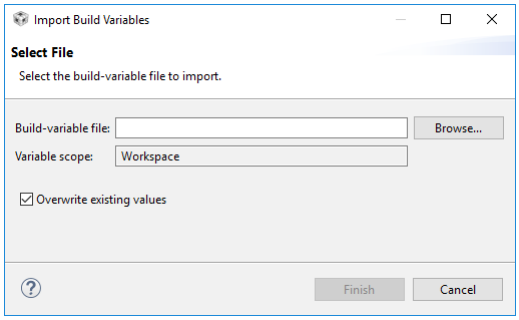
   Build variables for SwitcherWare:

```
[path] LibrarySwitcherWare = C:/Denkinetic/SwitcherWare/SwitcherWare_00_08_00
[path] C2000Ware = C:/ti/c2000/C2000Ware_3_04_00_00
```

   Build variables for SwitcherBlocksX2C:

```
[path] LibrarySwitcherBlocksX2C = C:/Denkinetic/SwitcherBlocksX2C/SwitcherBlocksX2C_00_01_00
[path] X2C_ROOT = C:/LCM/X2C/v6.3.2018
```

2. In CCS, select the menu **Window > Preferences**.
3. In the preferences window navigation pane, select **Code Composer Studio > Build > Variables**.
4. Click on the **Import...** button to show the **Import Build Variables** dialogue box.



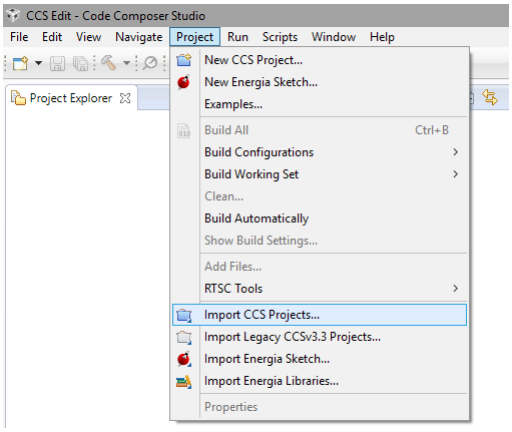
1. Click on the **Browse** button and select the vars.ini file.
2. Select the check-box to **Overwrite existing values**.

Import a CCS Project

The easiest way to start a new CCS project for SwitcherGear is to copy an example project supplied by Denkinetic. Example projects for the SwitcherBlocksX2C Library are located in the examples folder in the install package. Or a custom project for your application has been supplied directly to you.

Import a copy of an existing project into the workspace.

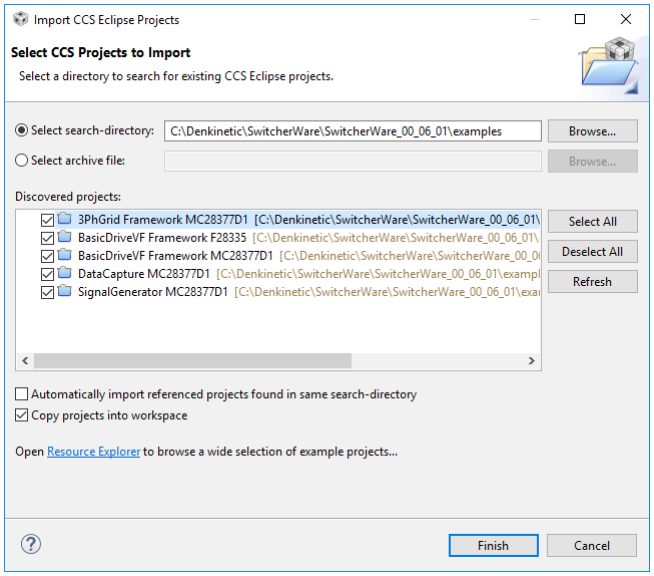
1. In CCS, select the menu **Project > Import CCS Projects…** .



2. Select the radio button for **Select search-directory**.

   If a project has been supplied in a .zip archive file, you can select the radio button for **Select archive file**.

3. Click the **Browse** button at the right side and navigate to the folder that contains the example CCS project (or .zip archive file).



4. In the list of discovered projects, select the check-box(es) for the project(s) you want to copy.
5. Select the check-box to **Copy projects into workspace**.
6. Click the **Finish** button.

IMPORTANT	The import process may fail if the project was created with a version of CCS that is newer than the version of CCS that is installed on your system. This is because the project relies on resources that your version of CCS does not have. In this case, you should update CCS to the latest version.

Version Control

You should consider using version control for your code projects. CCS has built-in support for git.

Using Scilab / Xcos

Scilab includes the Xcos graphical modelling environment, which has similar features and look/feel to other graphical modelling environments.

Xcos Model Files

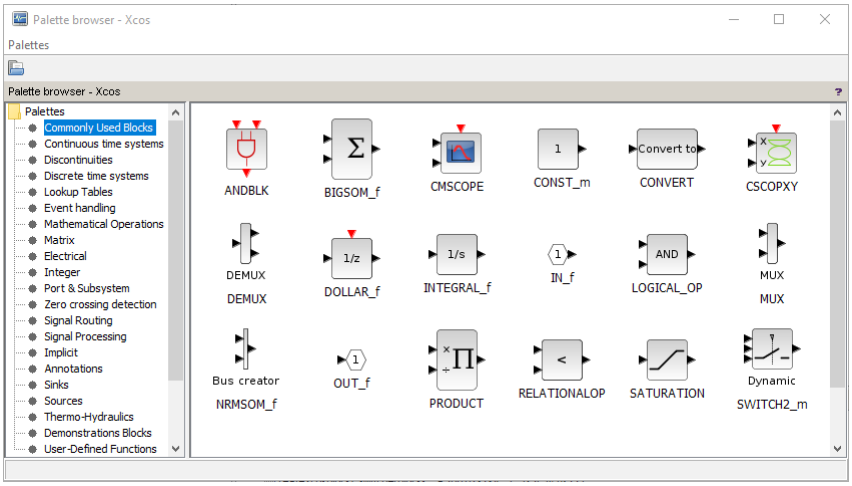
You can create a new Xcos diagram by selecting the menu item **Applications > Xcos**. Make sure that the correct menu is available by first clicking on the Scilab 5.5.2 Console pane.

When using X2C to generate control code, you should consider starting from a template file supplied by Denkinetic. A template file contains all the required blocks and is ready to accept your control model. In contrast, a new Xcos file must be manually set up with IN and OUT blocks that match the separate microcontroller application.

Xcos Palette Browser

The **Palette Browser** window is shown automatically when you open an Xcos window. Alternatively, you can open the palette browser window by selecting the **Palette Browser** item from the **View** menu.

The Palette Browser contains blocks that you can use to build Xcos models. The blocks are grouped into categories of similar blocks. The standard Xcos blocks are grouped into a top level section called **Palettes**.

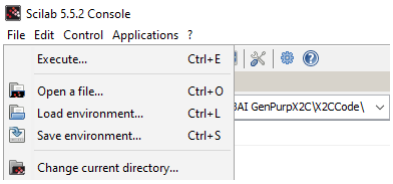


Scilab Scripts

Scilab script files contain Scilab commands and have a filename suffix `.sci` or `.sce`.

Running a script from Scilab

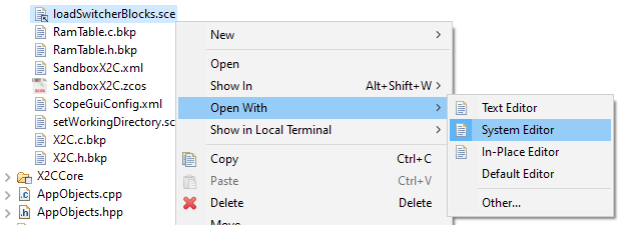
To run a script from Scilab, select the menu item **File > Execute...** and use the file browser to navigate to select the script file.



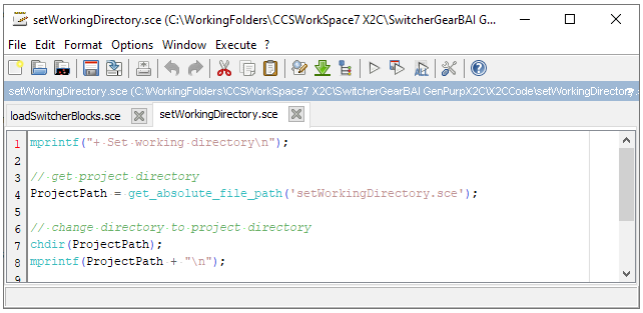
Running a script from CCS

You can also run a script from other locations, such as CCS.

1. In CCS, identify the script file (or a link to the script file) in the Project Explorer pane.
2. Right-click the script file to open the contextual menu, then select **Open With > System Editor**. This directs the operating system (Windows) to open the file using the default application – Scilab.



3. The script file will be opened in Scilab SciNotes text editor. Note that the editor window may be hidden behind other windows, so you may need to minimise other windows or click on its button in the task bar to bring it to the front.
4. In SciNotes, execute the script file using one of these methods:
   a. Press key **F5**; or
   b. Click on the Execute toolbar button; or
   c. Select menu item **Execute > Save and execute**.



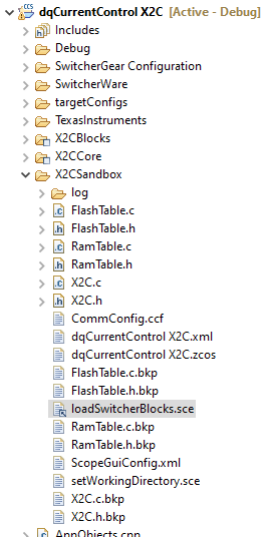
Starting Scilab

Loading SwitcherBlocksX2C

SwitcherBlocksX2C is a custom X2C library, which is not loaded with the standard X2C libraries when Scilab is run.

A Scilab script for loading SwitcherBlocksX2C is provided, and must be executed each time Scilab is run. The script is called **`loadSwitcherBlocksX2C.sce`** and is located in the SwitcherBlocksX2C install folder.

A link to the script is also available in the X2CSandbox folder of a SwitcherGear X2C CCS project. See page 14 for instructions to execute this script from CCS.



Setting the Scilab working directory

X2C requires the Scilab working directory to be set to the folder that contains X2C project file.

A Scilab script for setting the working directory is provided in the X2CSandbox folder of each SwitcherWareX2C CCS project. The script is called **`setWorkingDirectory.sce`**. See page 14 for instructions to execute this script from CCS.

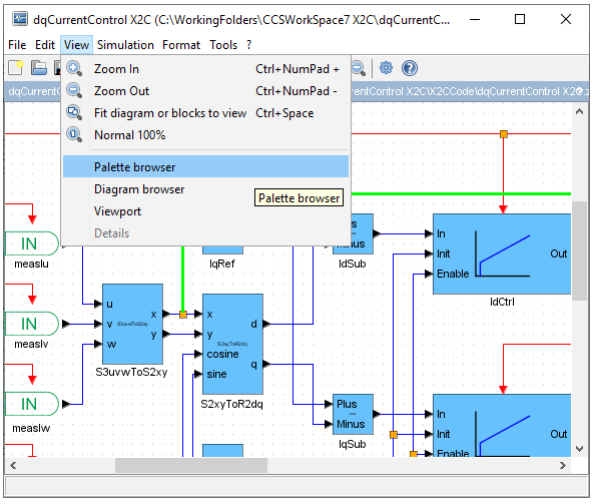
Opening X2C project file (existing project, SwitcherGear template, blank Xcos)

An X2C project file is a Scilab/Xcos diagram file that contains a graphical model built from X2C blocks. The filename ends in .zcos.

The X2C project file for a SwitcherGear X2C CCS project is located in the X2CSandbox folder of the CCS project. To open the X2C project file from CCS, right-click the file to open the contextual menu, then select **Open With > System Editor**.

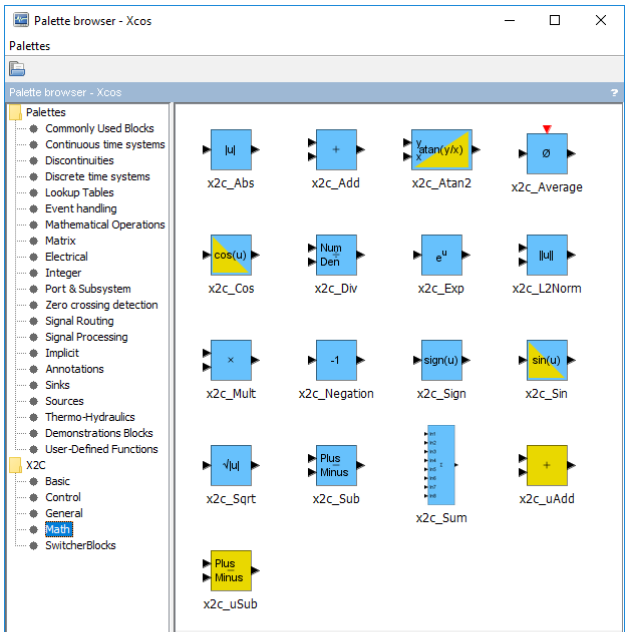
Showing the Palette Browser

From the menu bar of the X2C project file, select **View > Palette browser**.



If you have successfully installed X2C, the Palette browser contains a section called **X2C**. The free version of X2C includes block palettes called Basic, Control, General and Math. This section should also contain a block palette called SwitcherB-

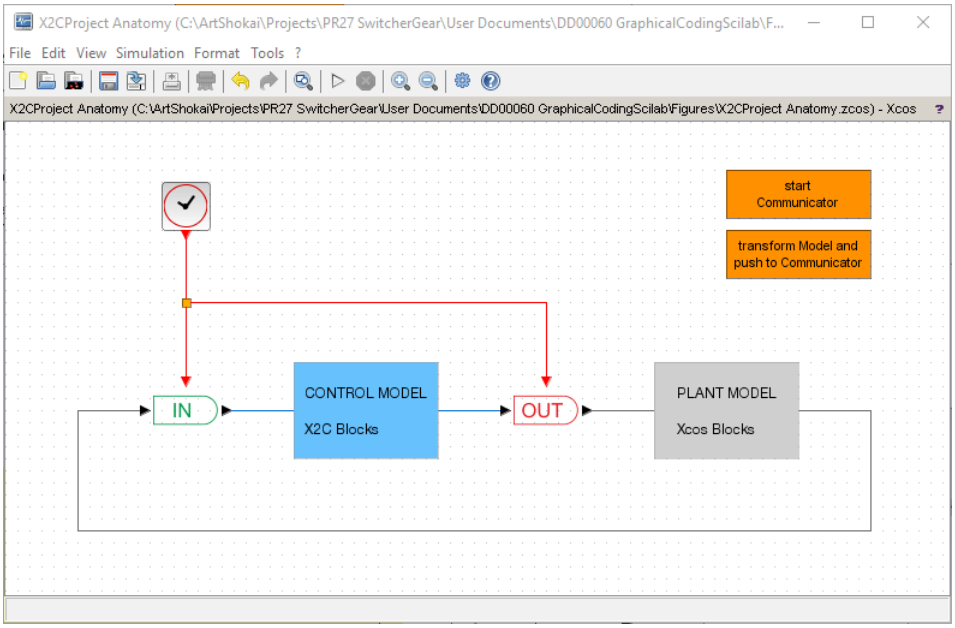
locksX2C. If none of these palettes is listed below the section heading **X2C**, the section may be collapsed – double-click on the section heading to reveal and collapse the section.



X2C Projects

An X2C project file is a Scilab/Xcos diagram file that contains a graphical model built from X2C blocks. The filename ends in .zcos.

The following figure shows the general structure of a X2C project. The graphical model directly resembles a control system block diagram, where the output of the controller drives the plant, and the state of the plant is measured and fed back as the input to the controller.



The X2C project can be used in two ways:

1. **Real-Time Target** The control model is converted to C code and is executed in real-time on the SwitcherGear controller hardware. The *real-time* control model controls a *physical* plant (e.g. a motor, a 3-phase grid, etc.) and the plant model is ignored.
2. **Off-line Simulation** The control model and the plant model are both executed in the Scilab/Xcos simulation environment. The *simulated* control model controls the *simulated* plant model.

IN & OUT Blocks

In the X2C system, the control model is bounded by green IN blocks and red OUT blocks. When the control model is executed on a real-time target, signals measured from the plant appear at the output terminals of the IN blocks. Likewise, signals presented by the control model to the input terminals of the OUT blocks are dispatched to the plant. When the control model is executed in an off-line simulation, the IN and OUT blocks pass input signals directly to their outputs.

The IN and OUT blocks are in the General palette in the X2C section of the Palette Browser.

The IN and OUT blocks are connected to a low-level framework application that interacts with the target microcontroller and hardware. Only advanced users should add or remove IN and OUT blocks in a X2C project. For other users, Denkinetic can supply a template X2C project that is matched to a SwitcherGear controller.

Control Model

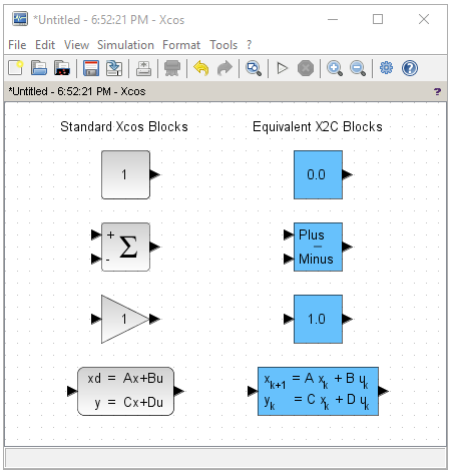
The control model is a graphical model that implements your control algorithm.

The control model is built by placing signal blocks and connecting them together with signal wires.

X2C Blocks

The control model is built mostly with X2C signal blocks, which are the blocks in the X2C section of the Palette Browser. Except for those listed below, you cannot use the standard Xcos signal blocks because they do not contain the resources that enable code generation and functionality with the X2C applications.

The X2C block libraries contain a wide variety of signal blocks that can be used to build most control systems. The figure below shows some typical Xcos signal blocks (grey) and their equivalent X2C signal blocks (blue).



X2C Angle Blocks

Most of the X2C blocks are coloured light blue. These blocks handle signals in a standard way.

Some X2C blocks have yellow highlighting that indicates that the block handles angle signals. The highlighting may cover the entire block, in which case all signals are angle signals. Otherwise, the position of the highlighting indicates the block terminals that carry angle signals. These blocks handle angle signals differently to standard signals to ensure they are limited to the range (-π, π]. This is important to prevent numerical problems, especially for operations such as addition, subtraction, scaling, etc. The following table shows the operations that have a standard signal block and a dedicated signal block for angular signals. Note that these blocks call the C library function **`fmod()`**, which can be computationally expensive.

Operation	Standard Block	Angle Block
Add two signals	**`X2C_Add`**	**`X2C_uAdd`**
Subtract one signal from another	**`X2C_Sub`**	**`X2C_uSub`**
Scale a signal by a constant factor	**`X2C_Gain`**	**`X2C_uGain`**
Limit the rate of change of a signal	**`X2C_RateLimiter`**	**`X2C_uRateLimiter`**
Integrate a signal with respect to time	**`X2C_I`**	**`X2C_uI`**

You must not use the angle blocks for non-angle signals.

Xcos Blocks

There is a small number of standard Xcos blocks that can be used in a X2C project. These are shown in the following table.

Xcos Palette	Block	Description
Event handling	CLOCK_c	Control timebase clock.
	freq_div	Divide a clock signal to execute blocks at a slower rate.
	CLKFROM, CLKGOTO	Tags for connecting clock events without clutter.
Port & Subsystem	SUPER_f	Super-block container for making hierarchical models.
	IN_f, OUT_f	Signal terminals for super-blocks.
	CLKINV_f, CLKOUTV_f	Clock event terminals for super-blocks.
Annotations	TEXT_f	A multi-line text box.

SwitcherBlocksX2C

Denkinetic has created custom X2C blocks for use with power converters.

The SwitcherBlocksX2C library must be manually loaded after starting Scilab using the script as described in the section Loading SwitcherBlocksX2C on page 14. After loading, the blocks are available in the Palette Browser.

Xcos Palette	Block	Description
SwitcherBlocks	x2c_S3uvwToS2xy x2c_S2xyToS3uvw x2c_S2xyToR2dq x2c_R2dqToS2xy	Coordinate transformations between various reference frames, including stationary 3-axis (UVW), stationary 2-axis (x/y or α/β) and rotating 2-axis (d/q). The block names use the abbreviations S3uvw, S2xy and R2dq, respectively, for these coordinate systems.
	x2c_SinCos	Derives the sine and cosine signals needed by the R2dq blocks.

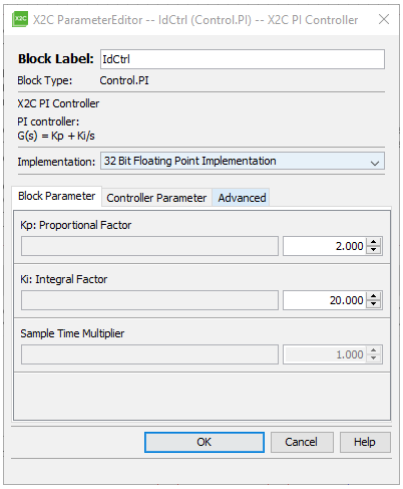
Block Implementation

X2C signal blocks usually have multiple internal implementations to cater for different numerical types (fixed-point or floating-point, with 8, 16, 32 or 64 bits). After placing the block, you must double-click and choose the correct implementation from the drop-down list. You must ensure when connecting blocks together that the numerical types are the same, or you must use conversion blocks to change between numerical types.

The choice of numerical type for the signals in your model depends on various factors. If you intend to create real-time control applications, you should use only numerical types that have hardware support in the target microcontroller. For SwitcherGear controllers, you should choose 32-bit floating-point for control signals.

Block Parameters

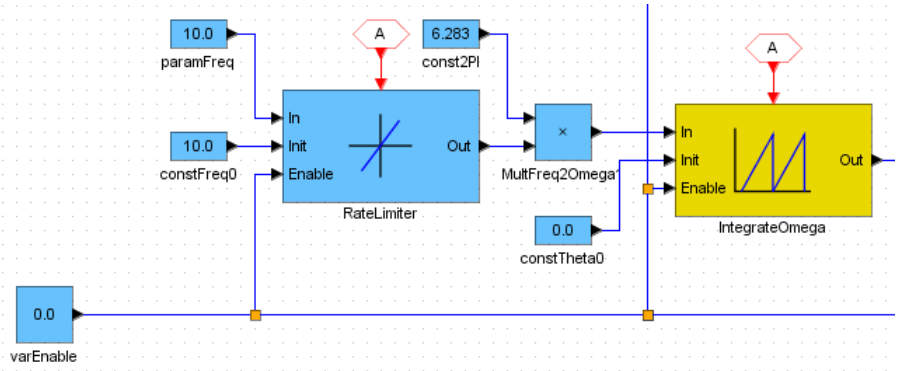
Some blocks have configurable parameters, for example a PI control block has configurable gains. After placing the block, you must double-click and set values for the parameters.



IMPORTANT	Controller gains and any other critical block parameters should be set to safe, initial values. You can tune them later.

Naming

You should name all blocks that you might need to identify later. You will need to identify blocks by name if you want to modify the block parameters in X2C Communicator, or plot the input/output signals in X2C Scope.



In particular, you should name the following blocks:

- x2c_Constant blocks that are used to set demand values or initial values for your control model.
- x2c_Constant block that is used to enable and disable the operation of the control model.
- Control blocks where you will modify the controller gains.

Plant Model

The plant model is a graphical model for the plant to be controlled by the control model. A plant model is only required for off-line simulation. It can be omitted if you will execute the control model only on a real-time target.

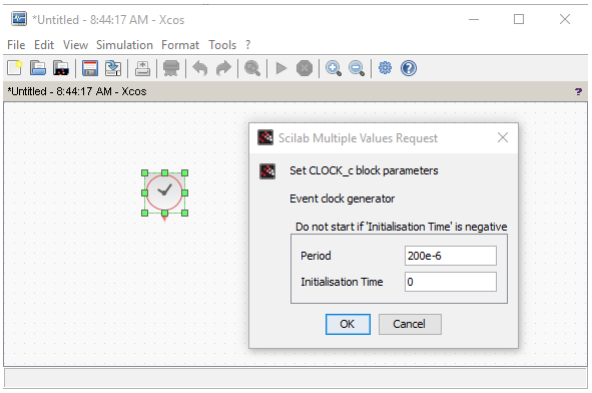
The plant model must be built with Xcos signal blocks, not X2C blocks.

The SwitcherBlocksX2C Library includes an Xcos file that contains some basic building blocks for power converters. Copy and paste them into your own plant models.

`<LibrarySwitcherBlocksX2C>/examples/Blocks for Power Converter Simulation.zcos`

Control Clock

There must be one clock in the X2C project file. The period of the clock can be edited by double-clicking on the block. However, the value set here does not affect the execution of the real-time target or the off-line simulation.



When the control model is executed on a real-time target, the execution period of the control model is determined by the framework application on the target.

When the control model is executed in an off-line simulation, the execution period of the control model is set by the Scilab variable **`X2C_sampleTime`**. You must set this variable so that X2C blocks can calculate internal variables that depend on the execution period. Set the value in the Scilab Console pane. The following example sets the execution period to 200 us, which corresponds to execution once per PWM cycle for a PWM switching frequency of 5 kHz.

```
-->X2C_sampleTime=200e-6
 X2C_sampleTime  =

    0.0002
```

The output of the clock is an event, not a signal. Note that in Xcos, the wiring and block terminals for events are red.

In the X2C project, you must connect a clock event to all blocks that have a red event input terminal.

X2C Application Blocks

You must place the following X2C blocks from the **X2C > Basic** palette into the X2C project:

- start Communicator
- transform Model and push to Communicator

These blocks automate the steps that are required when executing the control model on a real-time target.

Execution on a Real-Time Target

In this section, the graphical model for the controller is converted to C code and run as a real-time controller on the SwitcherGear hardware.

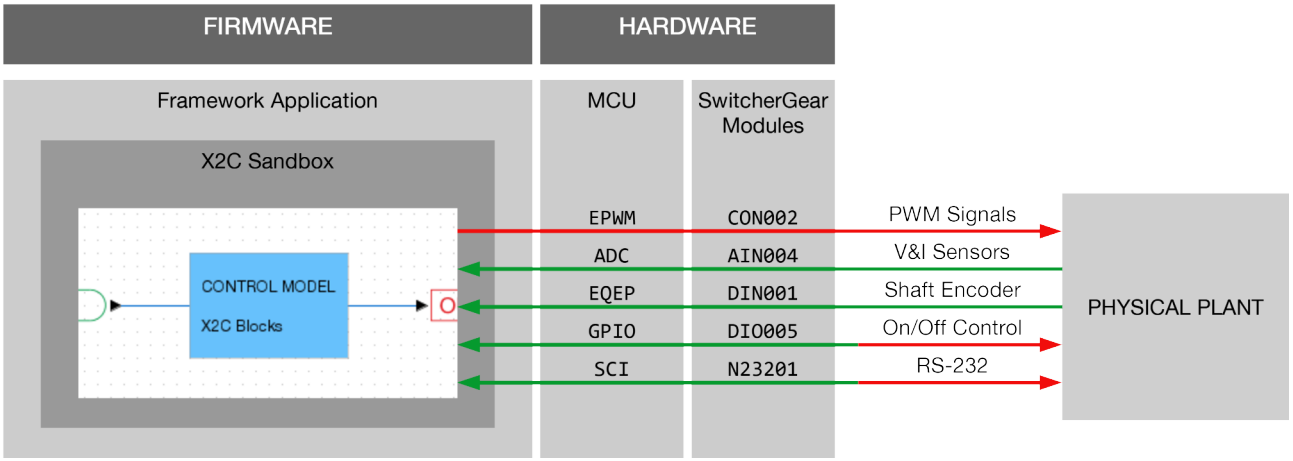
This section refers to material covered in previous sections.

Background

The real-time target contains the processor that executes the control model and the hardware interfaces that connect the control model to the physical plant. The SwitcherGear modular controller is a real-time target that is optimised for power converter applications.

The figure below shows:

- The X2C control model running inside the X2C Sandbox.
- The X2C Sandbox is hosted inside the framework application on the microcontroller.
- Hardware in the MCU and SwitcherGear modules transferring signals between the X2C control model and the physical plant.
- A physical plant, which may include IGBT converters, electrical machines, a 3-phase grid supply, voltage & current sensors, shaft encoders, contactors, switches, networks, other controllers, etc.



The framework application has these functions:

- Configure the low-level registers of the microcontroller.
- Periodically execute background tasks.
- Act as a bridge between the X2C Sandbox and the hardware.

The X2C sandbox has these functions:

- Make the measurements of the plant available to the control model. The measurements are pushed to the control model through the IN blocks in the X2C project.
- Dispatch the outputs of the control model to the plant. The outputs are pulled from the control model through the OUT blocks in the X2C project.
- Periodically execute the control model.

The X2C Sandbox must include the code resources to support the IN and OUT blocks in the X2C project file. This means that if a new IN/OUT block is added to the X2C project, the X2C Sandbox must be updated to support it. The signal for the IN/OUT block must be available in the framework application and/or the controller's hardware interfaces. Denkinetic

provides CCS framework applications with X2C Sandboxes that are configured to support various SwitcherGear controllers. Advanced users will be able to make customised changes to the X2C Sandbox and framework application.

Prepare

Install all required software.

In CCS, create a new workspace and set the build variables.

In CCS, import a copy of a X2C project into the workspace. The X2C project must be suitable for your SwitcherGear controller configuration and your intended physical plant. Contact Denkinetic if you need assistance.

Open X2C Project

Run the Scilab application.

Run the CCS application and launch the workspace that contains the X2C project.

From CCS, execute the Scilab script to load the SwitcherBlocksX2C library.

From CCS, execute the Scilab script to set the Scilab working directory.

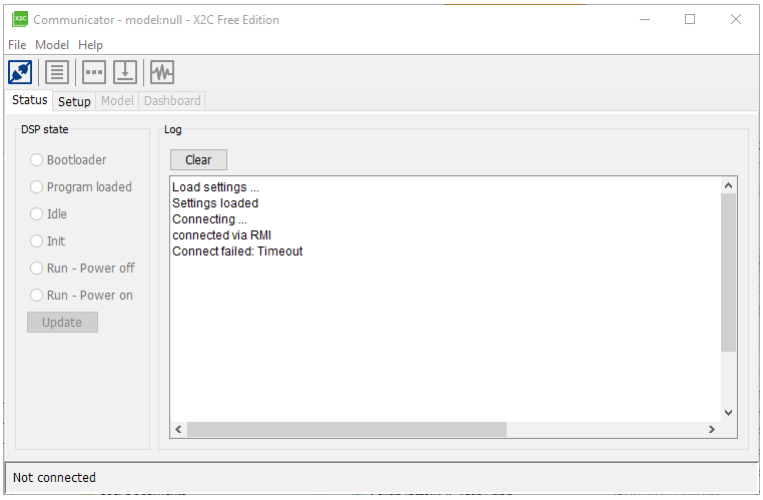
From CCS, open the X2C project file.

Edit Control Model

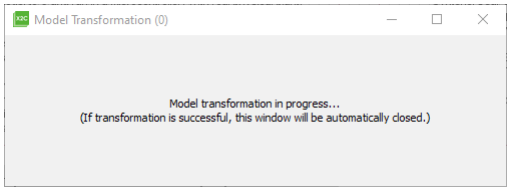
In Scilab, edit the control model in the X2C project. See section X2C Projects on page 17.

Set Control Model

In Scilab, launch X2C Communicator application by double-clicking the orange **start Communicator** block. The X2C Communicator window will open. It will attempt to make a serial connection to the target, but will fail if the control model is not already running on the target.

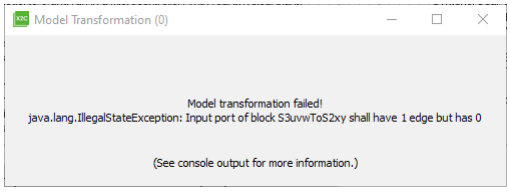


In Scilab, double-click the orange **transform Model and push to Communicator** block.

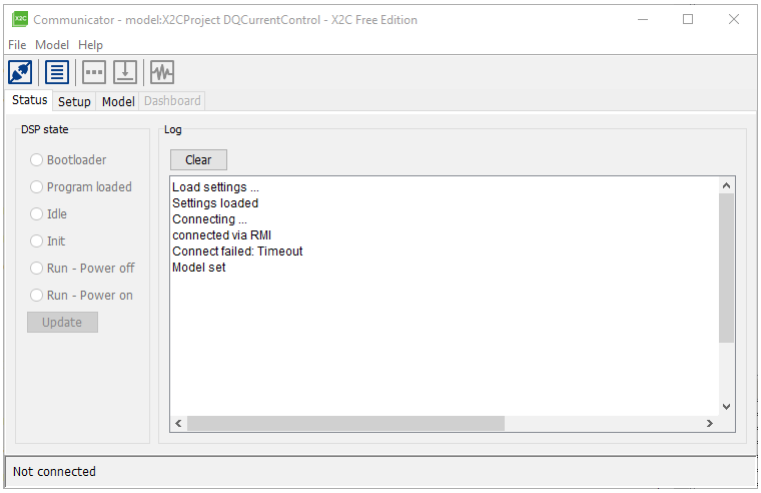


IMPORTANT	There must be no other open Xcos files when you perform this step.

If the transformation failed, you will receive an error message with a brief description of the problem. The console in the main Scilab window will contain further information that you should use to locate and rectify the problem.



If the transformation was successful, the X2C Communicator log will show that the control model has been set.

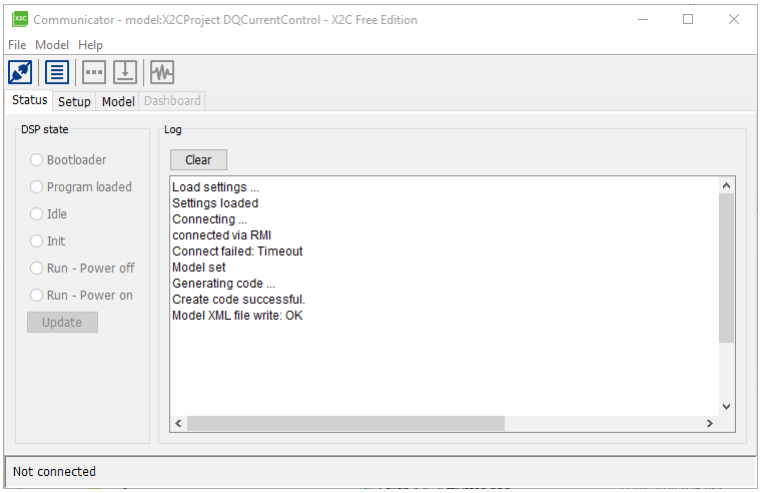


If you did not modify the control model, you can skip to the next section, Real-Time Interaction on page 27.

Create Code

In X2C Communicator, create the code for the model:

- Select the menu item **Model > Generate Code**, or
- Click on the **Create Code** button in the toolbar



The model will be converted into C source and header files, which will be created in the same folder as the X2C project file (the Scilab working directory).

Build and Flash Code

Connect the debug probe between the SwitcherGear controller and the PC. This provides a JTAG connection to the MCU for writing flash memory and performing low-level debugging.

Turn on the power to the SwitcherGear controller.

Turn OFF or disconnect all energy sources to the physical plant.

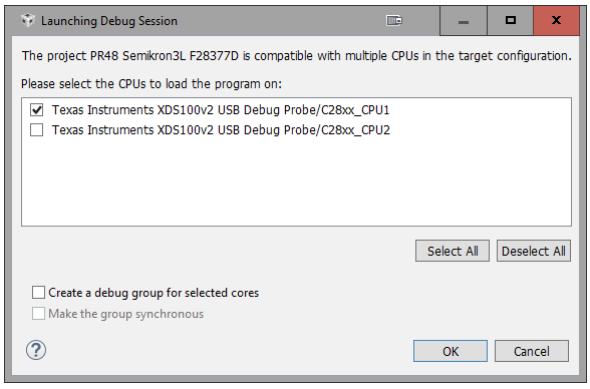
In CCS, open the file X2C.c, which can be found in the X2CSandbox folder of the project. Confirm that the date & time shown at the top of the file X2C.c agrees with the creation time at the step above.

In CCS, click the toolbar **Debug** button or press the **F11** key to debug the project. This performs the following tasks:

- compiles the project source files;
- links the project with the SwitcherWare Library and the SwitcherBlocks X2C Library;
- connects the debug probe to the MCU target;
- erases the flash memory of the MCU target;
- writes the build object into the MCU flash memory;
- starts execution of the MCU to run the C environment initialisation code; and
- pauses the program execution at the start of **`main()`**.

Any build errors are shown in the CCS **Console View**.

The first time that you build the project for a multi-core MCU, you will be prompted to select the CPUs to load the program on. Select **C28xx_CPU1** and deselect **C28xx_CPU2.**



During the above process, the view is changed from the Edit Perspective to the Debug Perspective, which shows information related to the debug process.

Execute the control model by clicking the toolbar **Resume** button or pressing the **F8** key.

The green **GRN** indicator on the front panel of the SwitcherGear controller will flash when the control model is running.

Real-Time Interaction

This section describes how you can interact in real-time with your control code that is running on the SwitcherGear controller. You will learn how to:

- Use a serial link to connect the X2C Communicator application to the control code.
- Change the values of model parameters in the control code.
- Use a virtual oscilloscope to observe signals in the X2C control model.

This section refers to material covered in previous sections.

Prepare

Set the control model in X2C communicator. See instructions in the previous section, Execution on a Real-Time Target.

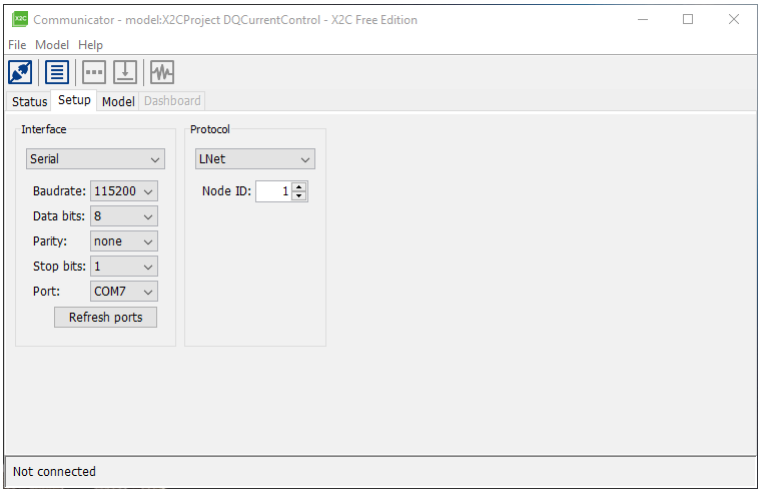
If you edited the control model X2C project file, you must also build and flash the code to the target.

Connect

Connect serial comms cable between the SwitcherGear controller and the PC. This provides a RS-232 connection to the application framework for interacting with the control model.

Turn on power to the SwitcherGear controller.

In X2C Communicator, select the **Setup** tab pane.



In the **Interface** group pane select **Serial** from the drop-down list and configure the following parameters:

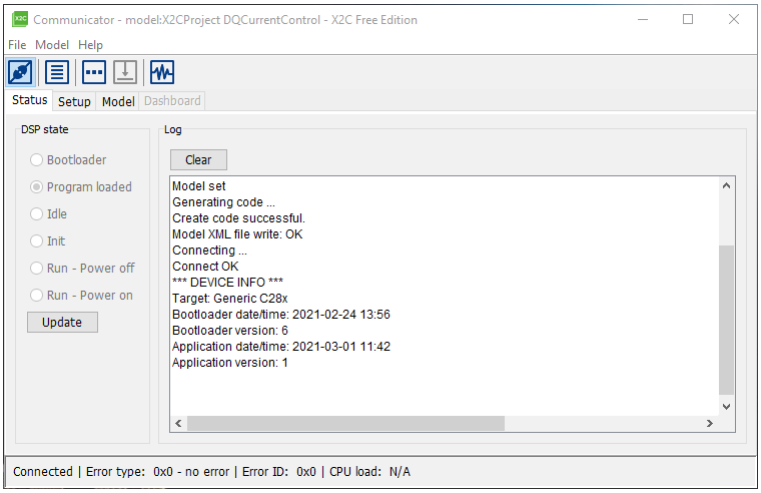
- Baud rate 115,200
- Data bits 8
- Parity none
- Stop bits 1
- Port Select your serial port – see Serial Port on page 7.

In the **Protocol** group pane select **LNet** from the drop-down list and configure the following parameters:

- Node ID 1

In X2C Communicator, select the **Status** tab pane.

Connect to the target by clicking the **Connect to Target** button in the toolbar.



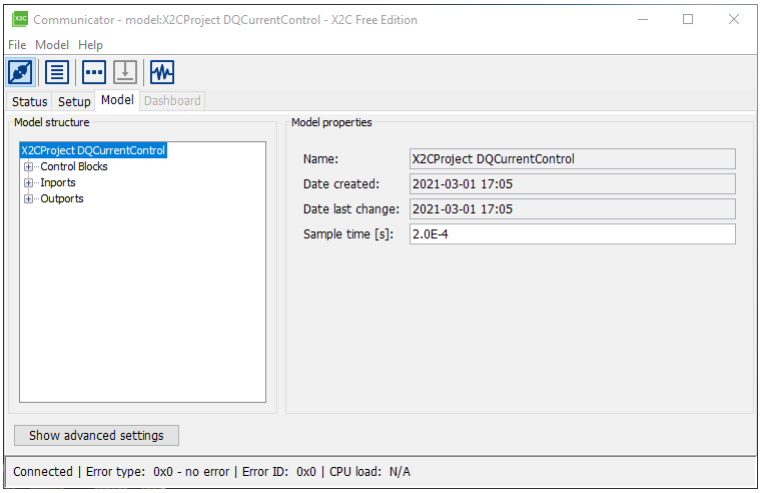
X2C Communicator log will show the connection status and some target information.

IMPORTANT	The RS-232 serial link that transfers the data is not robust against electrical interference. You must keep the cable away from sources of interference, especially the power stage and associated wiring in your experimental apparatus. A shielded cable may reduce interference.

Set Sample Time

The X2C sample time is the period at which the control model is executed. The control period is set in the application framework. X2C Communicator cannot extract this information automatically, so you must manually enter the value manually. This information is used for internal calculations that depend on the control period, such as integrator parameters and display of time scales on data plots.

In X2C Communicator, select the **Model** tab pane.



In the **Model Structure** list, select the name of the X2C Project.

In the **Model Properties** group pane, enter the value for the **Sample time** in seconds.

Calculate the Control Period

If you are unsure of the execution rate, you can use this relationship:

Control Period = PWM Period / Control Rate

Search the **`main.c`** file of the application framework for code similar to this:

```
MRE_ConverterHB3.Init(PERIPHERAL_ID_EPWM1, ControlRate_Single);
MRE_ConverterHB3.paramPWMFreq = 5000.0;
```

The second line sets the PWM frequency of a 3-phase converter object to 5 kHz, so the PWM period is 200 us.

The first line indicates that the control model is to be executed once per PWM period, i.e. the control rate is 1, so the control period is equal to the PWM period. An argument of **`ControlRate_Double`** would indicate that the control model is to be executed twice per PWM period, i.e. the control rate is 2, so the control period is equal to half the PWM period.

For more information, see the documentation in the SwitcherWare install folder for the **`HB3Sym_<MCU>`** class.

Modify Block Parameters

You can modify the parameters of model blocks while the control model is running.

WARNING	Carefully consider the effect of live changes before you make them.

There are two methods you can use to modify block parameters.

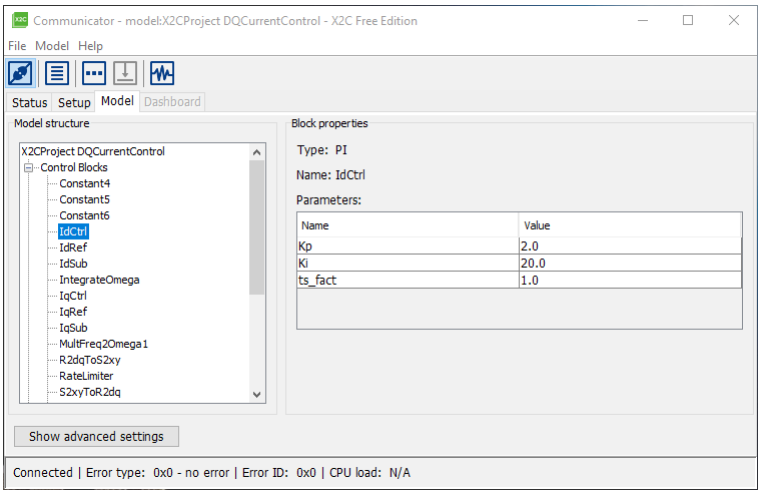
X2C Project File

In the X2C Project file, double-click on the block you want to modify.

Modifications made in this way will be preserved when the X2C Project file is saved.

X2C Communicator

In X2C Communicator, select the **Model** tab pane.



In the **Model Structure** list, expand the **Control Blocks** node to show the list of blocks used in the control model.

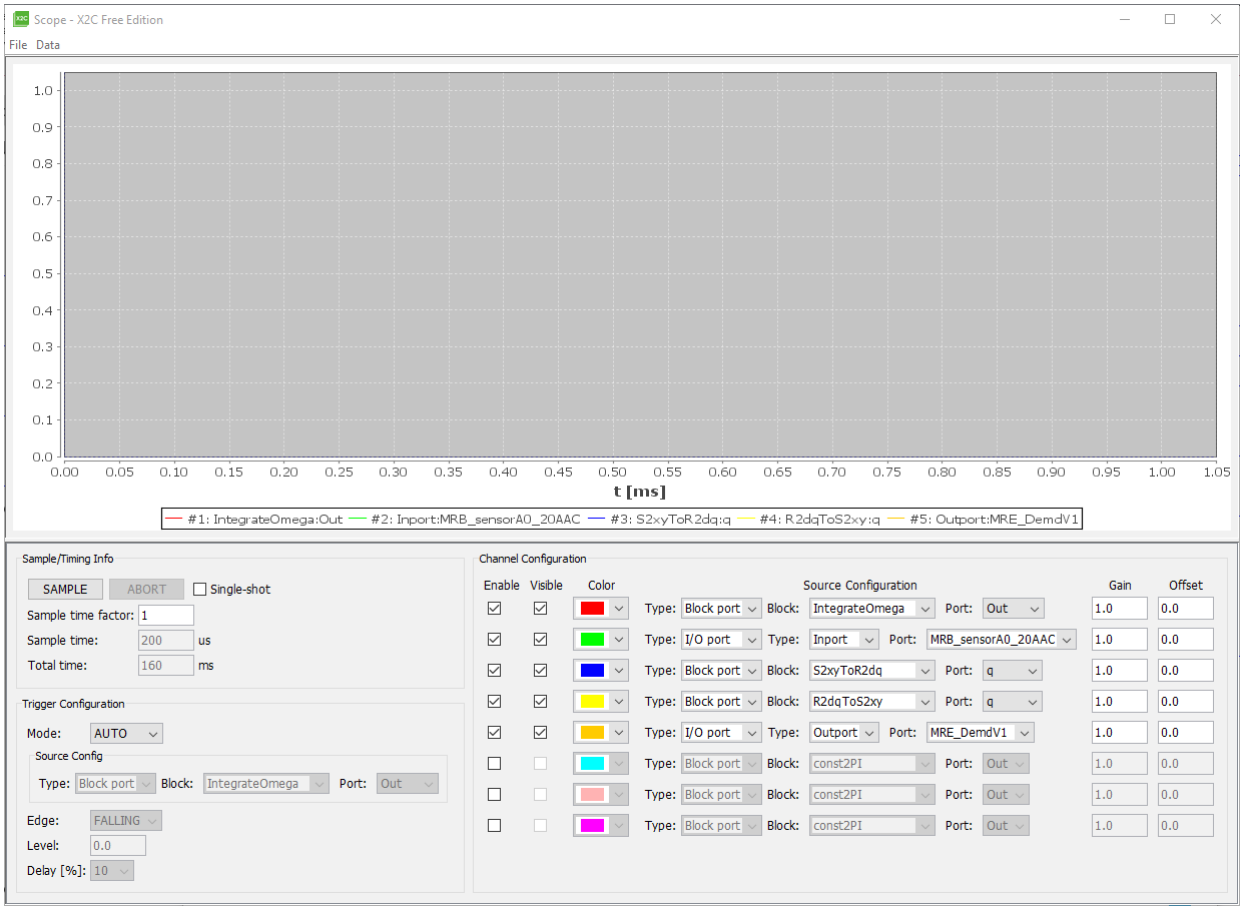
In the block list, select a block to show its properties in the **Block Properties** group pane.

Modifications made in this way will be lost when the session is ended.

X2C Scope

X2C Scope is a virtual oscilloscope that is part of the X2C installation. You can use it to capture signals from your control model, as you would with a conventional oscilloscope. The virtual scope has 8 signal channels and one trigger channel with edge trigger capability.

Open the X2C Scope by clicking the **Scope** button in the toolbar.



Sample/Timing

SAMPLE	Click button to enable sample mode.
ABORT	Click button to disable sample mode.
Single-shot	When checked, sample mode aborts after one sample is captured.
Sample time factor	The number of control periods between successive samples. Set to 1 to capture samples every control period. Use a higher number if you need to capture data over a longer time, but don't need to capture data at every time sample.
Sample time	Displays the sample time set in X2C Communicator, Model Properties.
Total time	Displays the time duration of captured samples. This depends on the amount of memory allocated for sample capture in the application framework, the number of signals sampled, the numerical type of the signals, the sample time and the sample time factor.

The application framework used for SwitcherGear sets a memory size for capturing samples of 8 k words (16-bit words are the basic unit of addressable data on C2000 MCUs). This amount of memory can capture 1000 samples on 4 channels that have 32-bit floating-point signals.

Trigger Configuration

Mode : AUTO	When sample mode is enabled, the scope starts a capture as soon as soon as the capture buffer is ready. The remaining trigger settings are ignored.
Mode : NORMAL	When sample mode is enabled, the scope starts a capture when the trigger configuration are met. When the capture is complete, the scope is armed and waits again for the trigger conditions.
Source Config	Use the drop-down lists to select the signal that will be used to trigger the capture.
Edge	Select the direction of change of the source signal that will trigger the capture.
Level	Enter the value of the signal that will trigger the capture.
Delay	Time shifts the trigger instant relative to the first sample of the capture. A value of zero causes the trigger instant to coincide with the first sample of the capture. A positive value delays the trigger instant to after the first sample, which has the effect of allowing you to capture samples before the trigger instant.

Channel Configuration

Enable	Enables capture of sample on the channel.
Visible	Displays the captured signal in the plot area.
Color	Select a colour for the channel plot.
Source Configuration	Use the drop-down lists to select the signal.
Gain, Offset	Display value = (Gain * Sample Value) + Offset

Other

Export the captured signal data to Matlab and Scilab formats for post-processing. Select the menu item **Data > Export**.

Take a screen shot for use in publications. Use the Windows Snip & Sketch application.

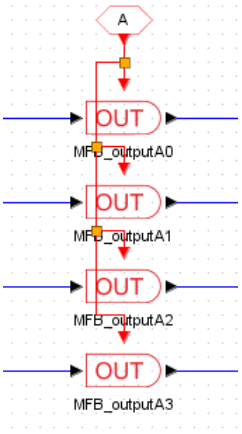
Be aware that there can be a significant delay in transferring captured data over the serial data link.

You can change the attributes of the plot area by right-clicking and selecting Properties.

SwitcherGear Analogue Outputs

If your SwitcherGear controller includes an analogue output module (Module AOV001) you can send 4 control signals to an oscilloscope as analogue signals. There is no delay in viewing these signals, which you may prefer to the X2C Scope.

The X2C Project file must include OUT blocks for this purpose (see image below), and the application framework and X2C Sandbox must also be coded for this feature.

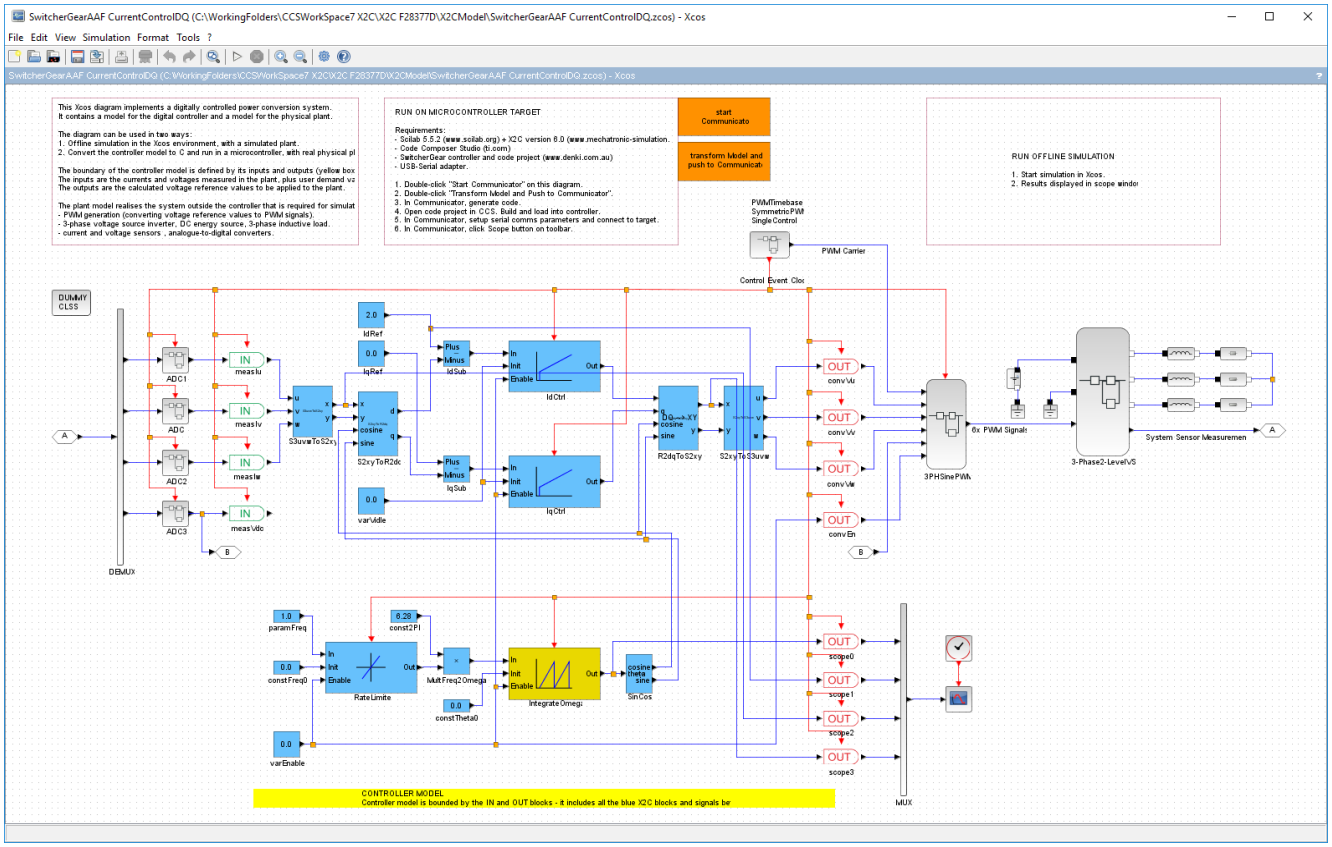


Simply connect the signals of interest with wire to the inputs of the OUT blocks. The output range of the module is ±10 V. You can add other blocks to scale and offset the signals, if required, to bring them into this range.

Off-line Simulation

In this section, a graphical model for the plant is developed in Scilab/Xcos. The model is run as a simulation on the PC.

This section refers to material covered in previous sections.



Controller Model

The controller model is implemented using X2C blocks.

This can be the same model that is used for real-time target execution.

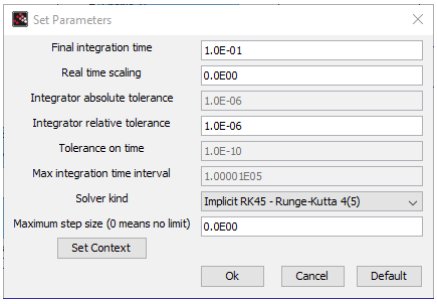
Plant Model

The plant model is implemented using standard Xcos blocks.

Scilab/Xcos provides Modelica blocks for modelling electrical systems, which are in the Electrical palette.

Set Parameters

Select menu item **Simulation > Setup**.



Enter the stop time for the simulation for the **Final integration time**.

To run the simulation, click on the toolbar **Start** button, or select menu item **Simulation > Start**.

Simulation results will be plotted for any Xcos Scope blocks that are present in the model.

Revision History

Revision	Date	Changes From Previous Release
1	3 Mar 2021	▪ Original release.