Simulink interface for real-time control of the RehaStim2 stimulator

Using the Embedded Coder with Eclipse IDE

Version 1.0

Table of Contents

1. Introduction

In order to allow the realisation of custom stimulation programs, a stimulator interface for Simulink was developed. This allows scientists to change stimulation parameters like pulse widths and current amplitudes in real-time using the ScienceMode2 protocol. Every pulse can be adjusted individually. Another interesting feature is the triggering of higher frequency pulse groups, like doublets and triplets. Communication between the stimulator and the PC is established via USB (virtual COM port). The communication link is Galvanically isolated for safety reasons.

Using the large block set available within Simulink, the user is able to create complex stimulation patterns. Even the set-up of state machines as well as the design and the implementation of feedback control systems are possible.

The interface consists of a Simulink block (see Figure 1) with underlying C s-function. Using the “Embedded Coder” together with the “Simulink Coder”, the diagram can be compiled and a soft real-time executable with high priority for the scheduler can be built by using the Eclipse IDE tool chain. Signals and parameters can still be monitored and changed respectively through the “External Mode Control Panel” of the “Simulink Coder” within Simulink. This way of using the diagrams works under Linux and MS Windows.

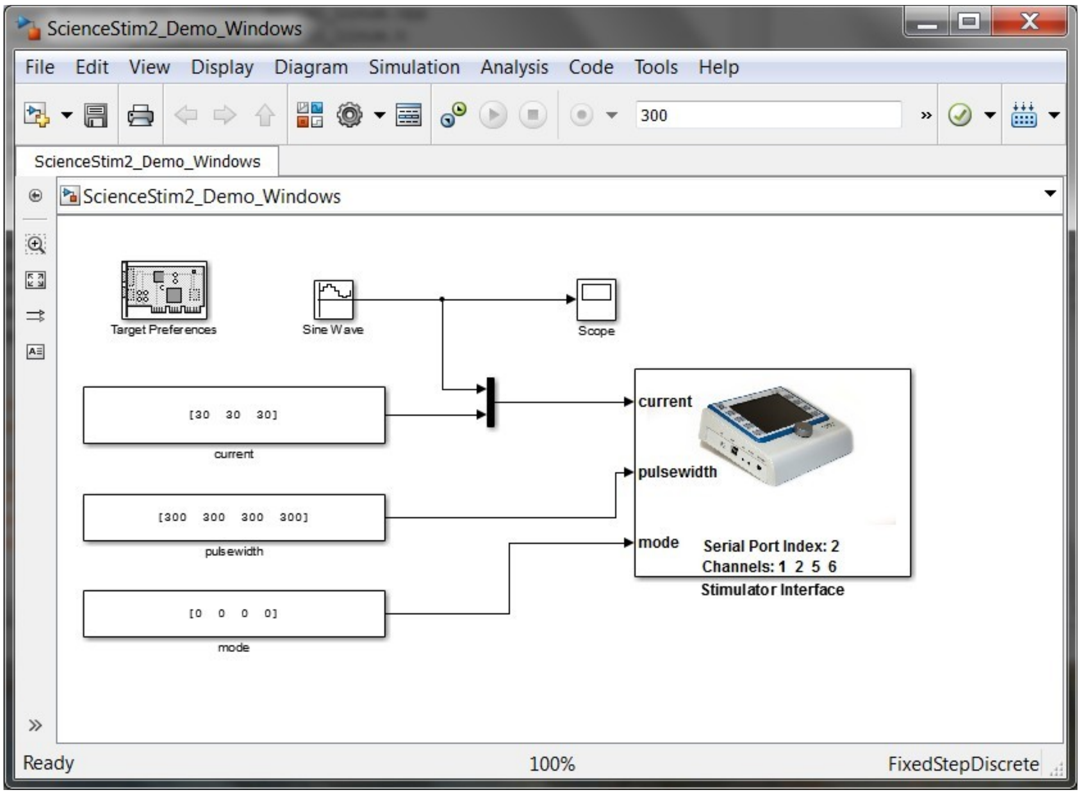


Figure 1: Simulink interface (4 channel stimulation example) under Linux for using the Eclipse IDE for code generation. Frequencies and update rate for the input signals are set via the block mask.

2. Requirements

2.1 Windows XP/Windows 7 (32/64 Bit)

- Matlab 2012a oder 2012b (32 Bit) with Simulink
- Simulink Coder, Matlab Coder, Embedded Coder
- Eclipse IDE for C/C++ Developers (Eclipse Ganymede Sr2 Packages)
- JRE 6.0 (Java version 1.6.x)
- MinGW - Minimalist GNU for Windows (including GCC, MSYS and GDB)
- Gnumex for compiling Matlab/Simulink mex files with GCC

2.2 Ubuntu 12.04 LTS (32 Bit)

- Matlab 2012a oder 2012b (32 Bit) with Simulink
- Simulink Coder, Matlab Coder, Embedded Coder
- Eclipse IDE for C/C++ Developers (Eclipse Ganymede Sr2 Packages)
- Sun JRE 6.0 (Java version 1.6.x)
- GCC and related tools

3. Installation of the stimulator interface

3.1 General information

Embedded Coder supports the Eclipse™ integrated development environment (IDE) for project generation and build automation. With Embedded Coder, you can generate Eclipse projects from your Simulink models and implement them on supported processors (Host PC (Intel/AMD)). The generated projects include application C source code files and assembler code files. Therefore only C s-functions can be used in the Simulink models. C++ s-functions are not supported so far.

However, the ScienceMode2 communication protocol as well as the serial communication are implemented in form of open source C++ classes (Gnu Public License - GPL). In order to use these classes through the stimulator interface C s-function it was necessary to create wrapper functions for all stimulator related class methods. A static library was built containing the C++ classes and wrapper functions. The wrapper functions are called within the C s-function after linking against this static library and against the standard GNU GCC C++ library.

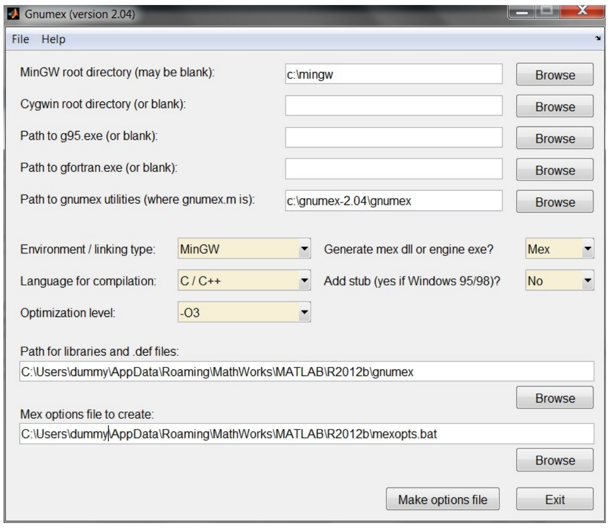
The following installation steps have to be carried out only once.

3.2 Windows XP/Windows 7 (32/64 bit)

1. Install the **32bit version of Matlab R2012a or R2012b** even if you run Windows 7 64 Bit. Therefore run the **setup.exe** in the **directory /bin/win32** of your Matlab installation media. In addition to Matlab you must install and activate **Simulink Coder, Matlab Coder** and **Embedded Coder.** Please note that you can install the 32 bit version of Matlab in parallel to the 64 bit version using two different directories.
2. Download the Windows 32 bit version of Eclipse IDE for C/C++ Developers (Eclipse Ganymede Sr2 Packages) from

   http://www.eclipse.org/downloads/packages/release/ganymede/sr2

   Install (unzip) it to c:/eclipse-cpp-ganymede-SR2-win32 (or any other location which you should remember).
3. Download MinGW – the Minimalist GNU for Windows – from http://www.mingw.org/. Install it in c:\MinGW and select during the installation process
   - C Compiler
   - C++ Compiler
   - MSYS Basis System
   7. MinGW Developer ToolKit
4. Add c:\MinGW\bin and c:\MinGW\include to the system search path.
5. Download Gnumex from http://gnumex.sourceforge.net/ (Version 2.04 was tested) and unzip the archive in c:\gnumex
6. Start Matlab (32 bit Version) and setup Gnumex by running the Matlab script **gnumex** inside the Gnumex directory c:/gnumex



Gnumex (version 2.04)

File Help

MinGW root directory (may be blank): c:\mingw

Cygwin root directory (or blank):

Path to g95.exe (or blank):

Path to gfortran.exe (or blank):

Path to gnumex utilities (where gnumex.m is): c:\gnumex-2.04\gnumex

Environment / linking type: MinGW

Generate mex dll or engine exe? Mex

Language for compilation: C / C++

Add stub (yes if Windows 95/98)? No

Optimization level: -O3

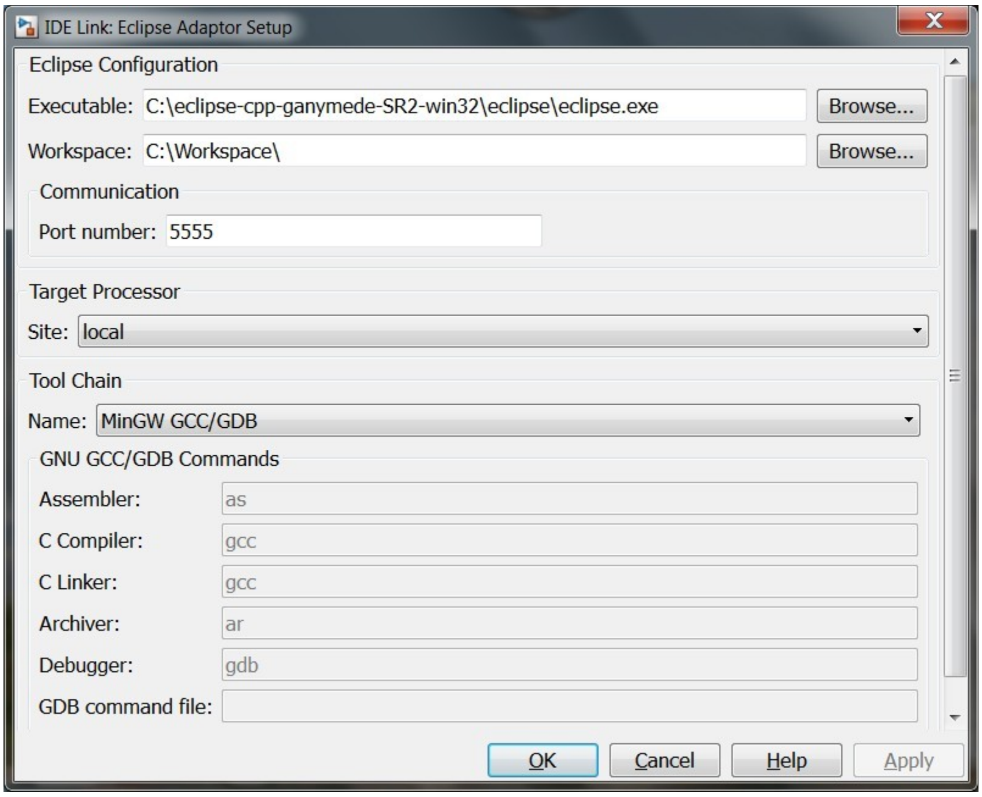
Path for libraries and .def files:
C:\Users\dummy\AppData\Roaming\MathWorks\MATLAB\R2012b\gnumex

Mex options file to create:
C:\Users\dummy\AppData\Roaming\MathWorks\MATLAB\R2012b\mexopts.bat

7. Setup the Eclipse IDE for Matlab by running the script **eclipseidesetup** inside Matlab.

   Specify where the eclipse executable is located.

   The executables generated from the simulink models can be found later in the specified **workspace directory**, e.g. c:\Workspace\



IDE Link: Eclipse Adaptor Setup

Eclipse Configuration

Executable: C:\eclipse-cpp-ganymede-SR2-win32\eclipse\eclipse.exe

Workspace: C:\Workspace\

Communication

Port number: 5555

Target Processor

Site: local

Tool Chain

Name: MinGW GCC/GDB

GNU GCC/GDB Commands

Assembler:	as
C Compiler:	gcc
C Linker:	gcc
Archiver:	ar
Debugger:	gdb
GDB command file:	

8. Create the ScienceMode2 directory which contains the source code for creating the static libraries and mex32 file of the C s-function related to the stimulator interface.

   Attention: The directory and the absolute path to the directory should not contain any blank characters!

   For example use **c:\ScienceMode2**

   Unzip the archive **Simulink_Sciencemode2.zip** inside ScienceMode2 directory.

9. Copy the static library **libstdc++.a** from **c:\MinGW\lib\gcc\mingw32\4.6.2\** to the ScienceMode2 directory.

10. Inside Matlab, change to the ScienceMode2 directory and run the script **compile_win** once to build the static library **libscm2.a** and **sfun_ScienceMode2.mexw32**

3.3 Ubuntu 12.04 LTS (32 bit/64 bit)

Note: The Eclipse IDE was successfully tested for 32 bit Linux. The 64 bit version could not be tested yet but should work.

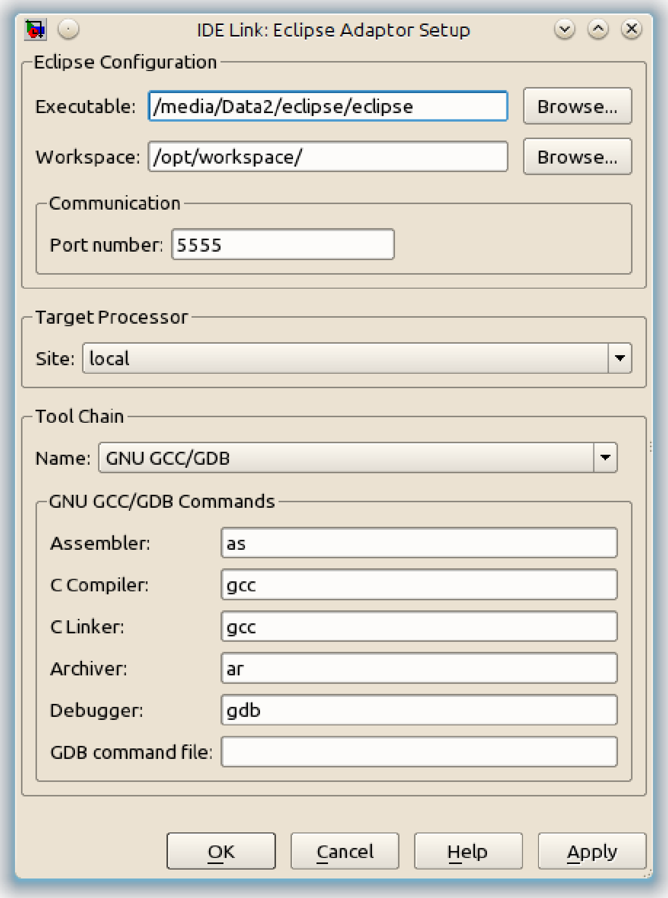
1. Install **Matlab R2012a** or **R2012b.** In addition to Matlab you must install and activate **Simulink Coder, Matlab Coder** and **Embedded Coder.**
2. Download the Linux 32 bit or 64 bit version (depending on your Linux) of Eclipse IDE for C/C++ Developers (Eclipse Ganymede Sr2 Packages) from

   http://www.eclipse.org/downloads/packages/release/ganymede/sr2

   Extract the archive somewhere and remember the location.
4. Install GCC and build tools as super user by using the command

   sudo apt-get install build-essential

   in a Linux terminal.
5. Setup the Eclipse IDE for Matlab by running the script **eclipseidesetup** inside Matlab. Specify where the eclipse executable is located.


IDE Link: Eclipse Adaptor Setup

Eclipse Configuration

Executable: /media/Data2/eclipse/eclipse

Workspace: /opt/workspace/

Communication

Port number: 5555

Target Processor

Site: local

Tool Chain

Name: GNU GCC/GDB

GNU GCC/GDB Commands

Assembler: as

C Compiler: gcc

C Linker: gcc

Archiver: ar

Debugger: gdb

GDB command file:

The executables generated from the Simulink diagrams can be found later in the specified **workspace directory**, e.g. \opt\workspace\

5. Create the ScienceMode2 directory which contains the source code for creating the static libraries and mexglx/mexglx64 file of the C s-function related to the stimulator interface.

   For example use **\home\dummy\ScienceMode2**

   Unzip the archive **Simulink_Sciencemode2.zip** inside ScienceMode2 directory.

6. Open a terminal. Change to the directory /usr/lib/ and locate with the command **find -name libstdc++.a** the static library **libstdc++.a**. Copy this library to the ScienceMode2 directory.

7. Inside Matlab, change to the ScienceMode2 directory and run the script **compile_linux** once to build the static library **libscm2.a** and **sfun_ScienceMode2.mexglx / sfun_ScienceMode2.mexglx64.**

4. Use of the stimulator interface

4.1 Setting up the stimulator

Connect the stimulator RehaStim2 to the PC using an USB cable. Start the stimulator and start the ScienceMode2 by selecting the programme “Science Mode 2”.

MS Windows: Please install the FDTI drivers coming with the stimulator. Before using Simulink please check in the Windows Systems Settings under Devices the assigned COM port of the stimulator. The COM port may change by connecting/disconnecting devices to the PC.

Ubuntu: Under Linux one can map the dynamically assigned port to a predefined fixed port like **/dev/stimulator**.

In a first step the serial number of the FTDI chip inside the stimulator must be determined. Therefore a terminal must be opened and the stimulator connected. The command **dmesg | tail** inside the terminal should then return with something like

```
[ 9479.712079] usb 3-1: new full-speed USB device number 4 using uhci_hcd
[ 9479.921799] ftdi_sio 3-1:1.0: FTDI USB Serial Device converter detected
[ 9479.921882] usb 3-1: Detected FT232RL
[ 9479.921888] usb 3-1: Number of endpoints 2
[ 9479.921894] usb 3-1: Endpoint 1 MaxPacketSize 64
[ 9479.921900] usb 3-1: Endpoint 2 MaxPacketSize 64
[ 9479.921906] usb 3-1: Setting MaxPacketSize 64
[ 9479.923784] usb 3-1: FTDI USB Serial Device converter now attached to
ttyUSB0
```

Here, **ttyUSB0** is the current dynamically assigned port name. The command **udevadm info --attribute-walk -n /dev/ttyUSB0 | sed -n '/FTDI/,/serial/p'** (please use the port name determined before) inside the terminal produces an answer like this:

```
ATTRS{manufacturer}=="FTDI"
ATTRS{product}=="FT232R USB UART"
ATTRS{serial}=="A4003OIf"
```

Here, **A4003OIf** is the serial number of the FDTI chip which we are interested in. After adding the line **SUBSYSTEMS=="usb", KERNEL=="ttyUSB*", ATTRS{serial}=="A4003OIf", NAME="sciencemode2", SYMLINK="hasomed_rehastim2, RUN+="bin/sh -c 'echo 2 > /sys/class/tty/ $KERNEL/device/latency_timer' ”** as super user to **/etc/udev/rules.d/10-udev.rules** the used stimulator (with the specified serial number) will always automatically be mapped to the port **sciencemode2** when connecting it to the PC. Additionally, the latency for sending data from the stimulator to the PC will be reduced to 2ms. If the file **10-udev.rules** does not exist then it must be created.

4.2 Setting up a new diagram

The directory for a new diagram must contain the following files

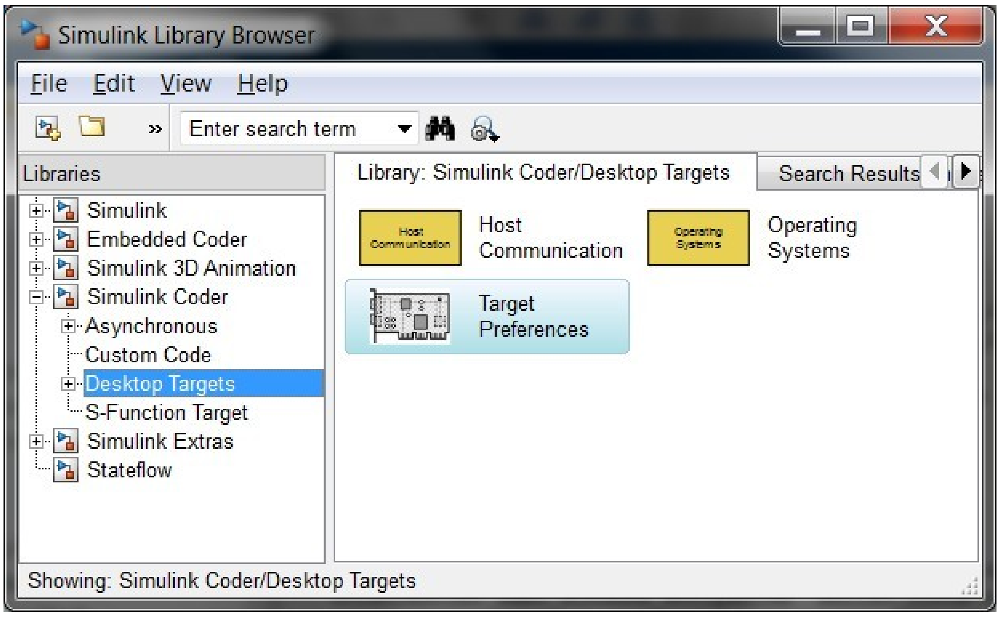
- **libscm2.a**
- **libstdc++.a**
- **sfun_ScienceMode2.mexw32 (MS Windows) or sfun_ScienceMode2.mexglx / sfun_ScienceMode2.mexglx64 (Linux)**
- **Sciencemode2_Lib.mdl**
- **Sciencemode2_help.html**
- **Rehastim-background.png**

Copy these files from the ScienceMode2 directory or add the ScienceMode2 directory to the search path of Matlab.

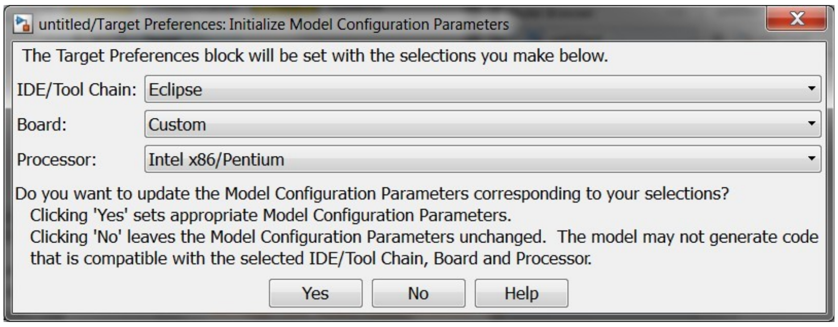
We highly recommend to use the demo Simulink diagrams
ScienceStim2_Demo_Windows / ScienceStim2_Demo_Linux
as templates for new diagrams.

In case of setting up a diagram from scratch the following steps have to be performed:

1. Create a new diagram
2. Insert the block **"Target Preferences"** from Simulink Coder/Desktop Targets in the diagram



3. In the Simulink diagram set the target preferences by clicking on the corresponding block as shown in the next three screen shots:


untitled/Target Preferences: Initialize Model Configuration Parameters

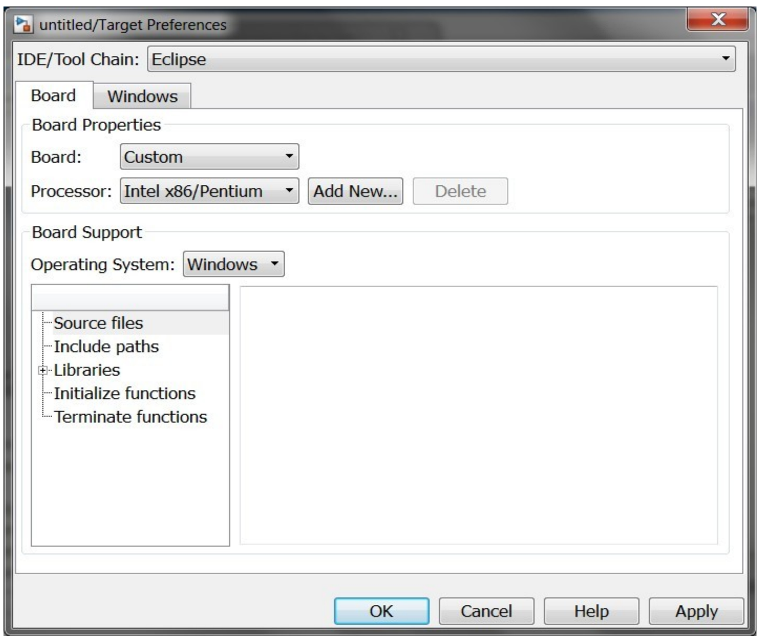
The Target Preferences block will be set with the selections you make below.

IDE/Tool Chain: Eclipse

Board: Custom

Processor: Intel x86/Pentium

Do you want to update the Model Configuration Parameters corresponding to your selections?
Clicking 'Yes' sets appropriate Model Configuration Parameters.
Clicking 'No' leaves the Model Configuration Parameters unchanged. The model may not generate code that is compatible with the selected IDE/Tool Chain, Board and Processor.



untitled/Target Preferences

IDE/Tool Chain: Eclipse

Board | Windows

Board Properties

Board: Custom

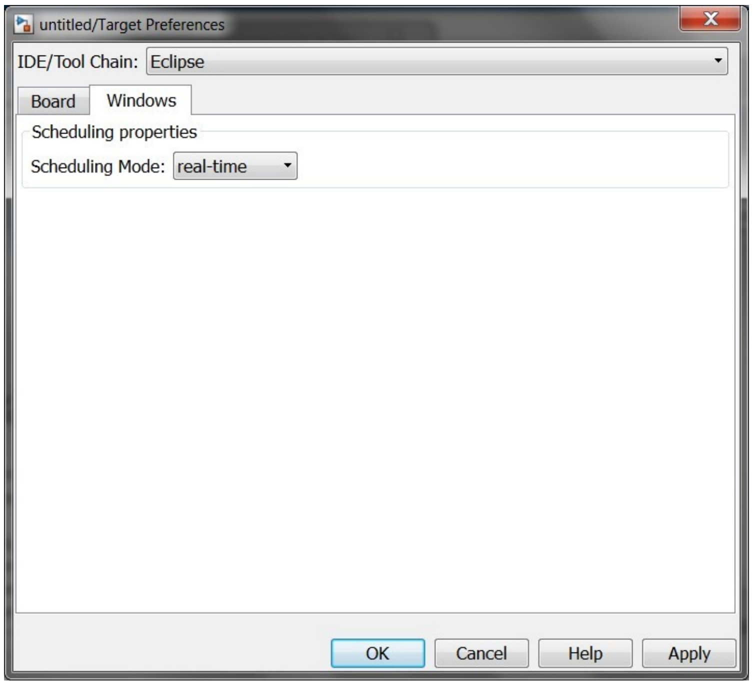
Processor: Intel x86/Pentium | Add New... | Delete

Board Support

Operating System: Windows

- Source files
- Include paths
- Libraries
- Initialize functions
- Terminate functions

OK | Cancel | Help | Apply



untitled/Target Preferences

IDE/Tool Chain: Eclipse

Board | Windows

Scheduling properties

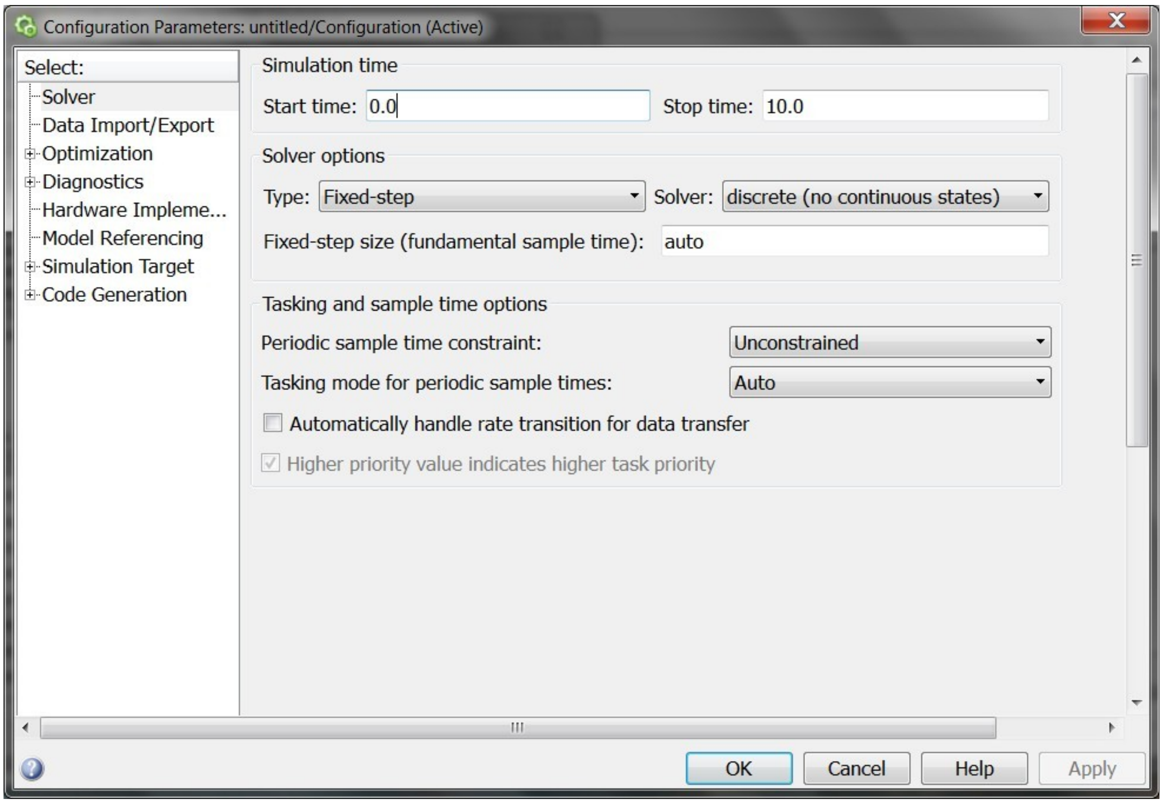
Scheduling Mode: real-time

OK | Cancel | Help | Apply

Please note: In Linux, select Linux instead of Windows during the configuration.

4. In the Simulink model please select **Simulation/Model Configuration Parameters**.

   Select under **Solver: Type: Fixed-Step, Solver: discrete**


Configuration Parameters: untitled/Configuration (Active)

Select:
- Solver
- Data Import/Export
- Optimization
- Diagnostics
- Hardware Impleme...
- Model Referencing
- Simulation Target
- Code Generation

Simulation time

Start time: 0.0 Stop time: 10.0

Solver options

Type: Fixed-step Solver: discrete (no continuous states)

Fixed-step size (fundamental sample time): auto

Tasking and sample time options

Periodic sample time constraint: Unconstrained

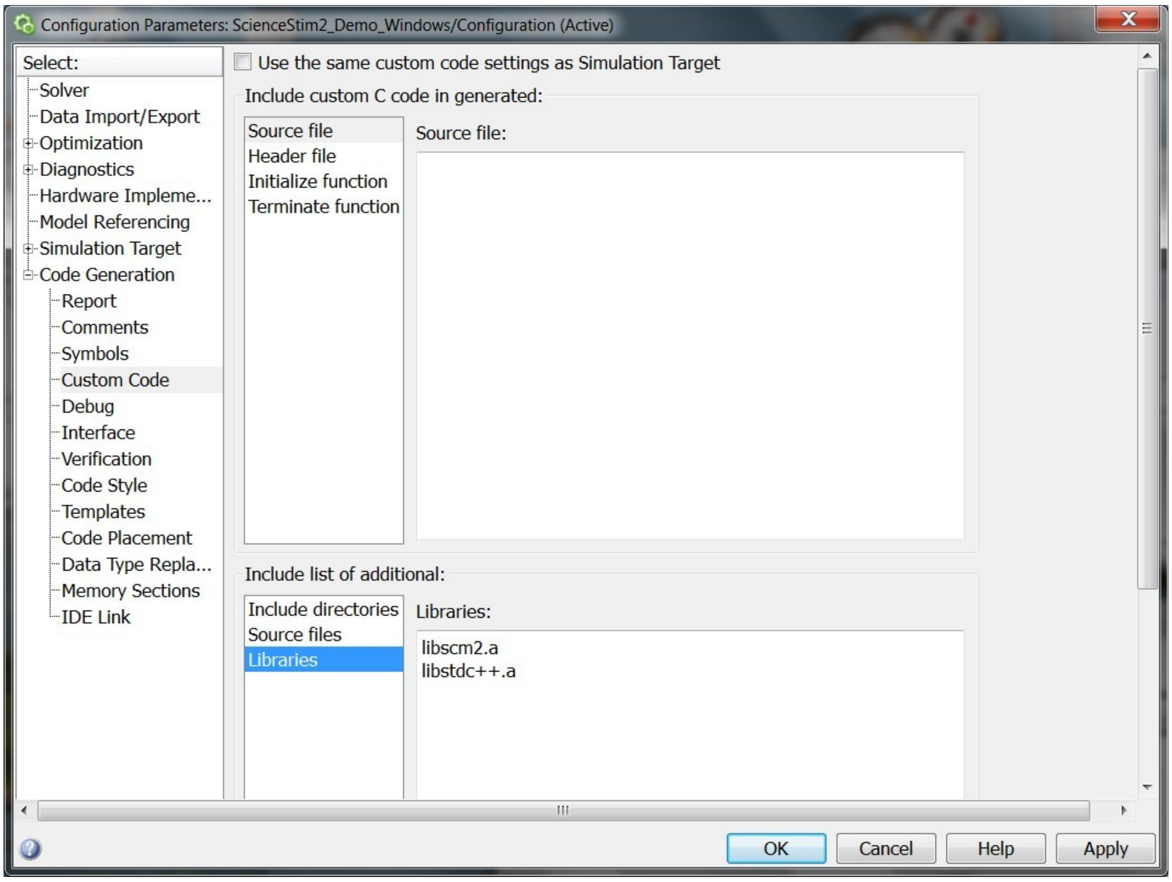
Tasking mode for periodic sample times: Auto

☐ Automatically handle rate transition for data transfer

☑ Higher priority value indicates higher task priority

OK Cancel Help Apply

Under **Code Generation/Custom Code:** Please add the static libraries **libscm2.a** and **libstdc++.a**


Configuration Parameters: ScienceStim2_Demo_Windows/Configuration (Active)

Select:
- Solver
- Data Import/Export
- Optimization
- Diagnostics
- Hardware Impleme...
- Model Referencing
- Simulation Target
- Code Generation
  - Report
  - Comments
  - Symbols
  - Custom Code
  - Debug
  - Interface
  - Verification
  - Code Style
  - Templates
  - Code Placement
  - Data Type Repla...
  - Memory Sections
  - IDE Link

☐ Use the same custom code settings as Simulation Target

Include custom C code in generated:

Source file / Header file / Initialize function / Terminate function

Source file:

Include list of additional:

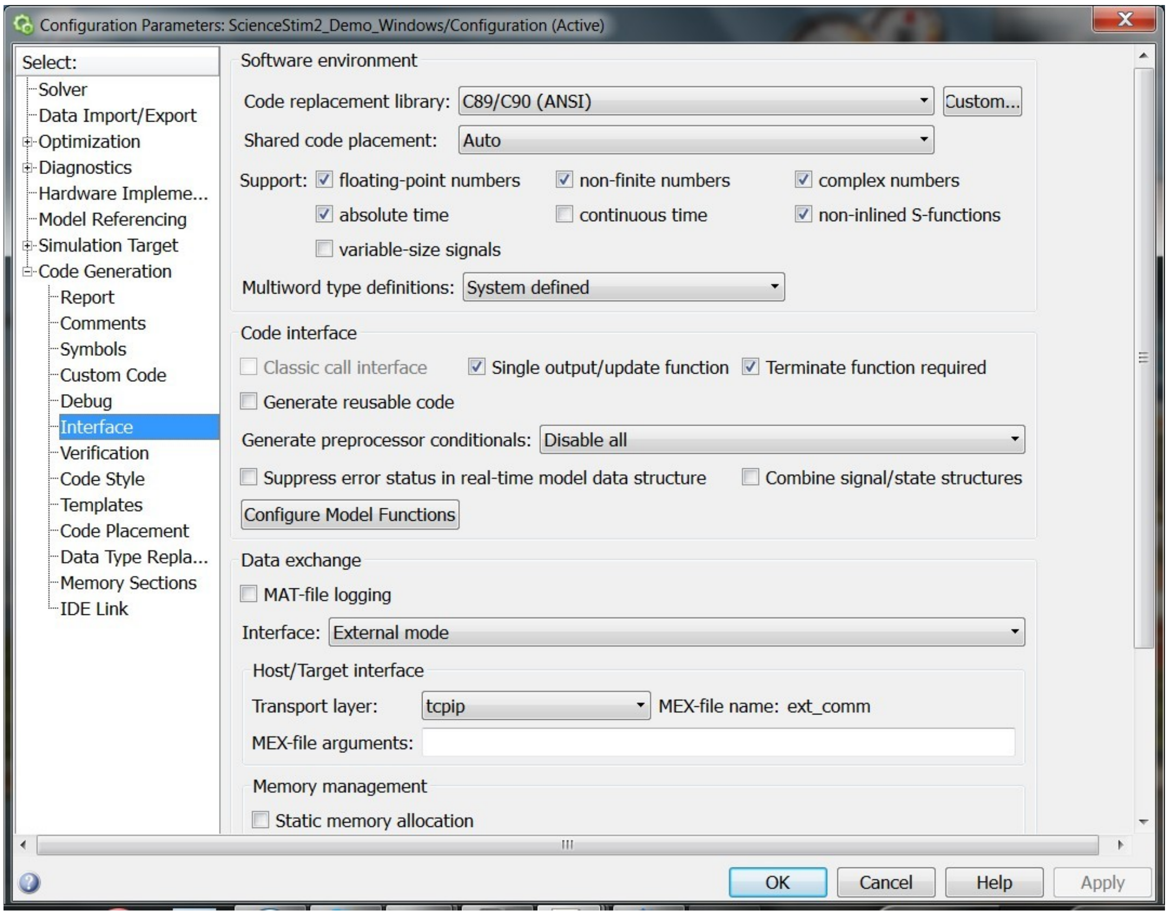
Include directories / Source files / Libraries

Libraries:

libscm2.a
libstdc++.a

OK Cancel Help Apply

Under **Code Generation/Inferface:** Enable **non-inlined s-functions** & choose the **External Mode Interface** for on-line monitoring signals/ changing parameters


Configuration Parameters: ScienceStim2_Demo_Windows/Configuration (Active)

Software environment

Code replacement library: C89/C90 (ANSI)

Shared code placement: Auto

Support: ☑ floating-point numbers ☑ non-finite numbers ☑ complex numbers ☑ absolute time ☐ continuous time ☑ non-inlined S-functions ☐ variable-size signals

Multiword type definitions: System defined

Code interface

☐ Classic call interface ☑ Single output/update function ☑ Terminate function required

☐ Generate reusable code

Generate preprocessor conditionals: Disable all

☐ Suppress error status in real-time model data structure ☐ Combine signal/state structures

Data exchange

☐ MAT-file logging

Interface: External mode

Host/Target interface

Transport layer: tcpip MEX-file name: ext_comm

MEX-file arguments:

Memory management

☐ Static memory allocation

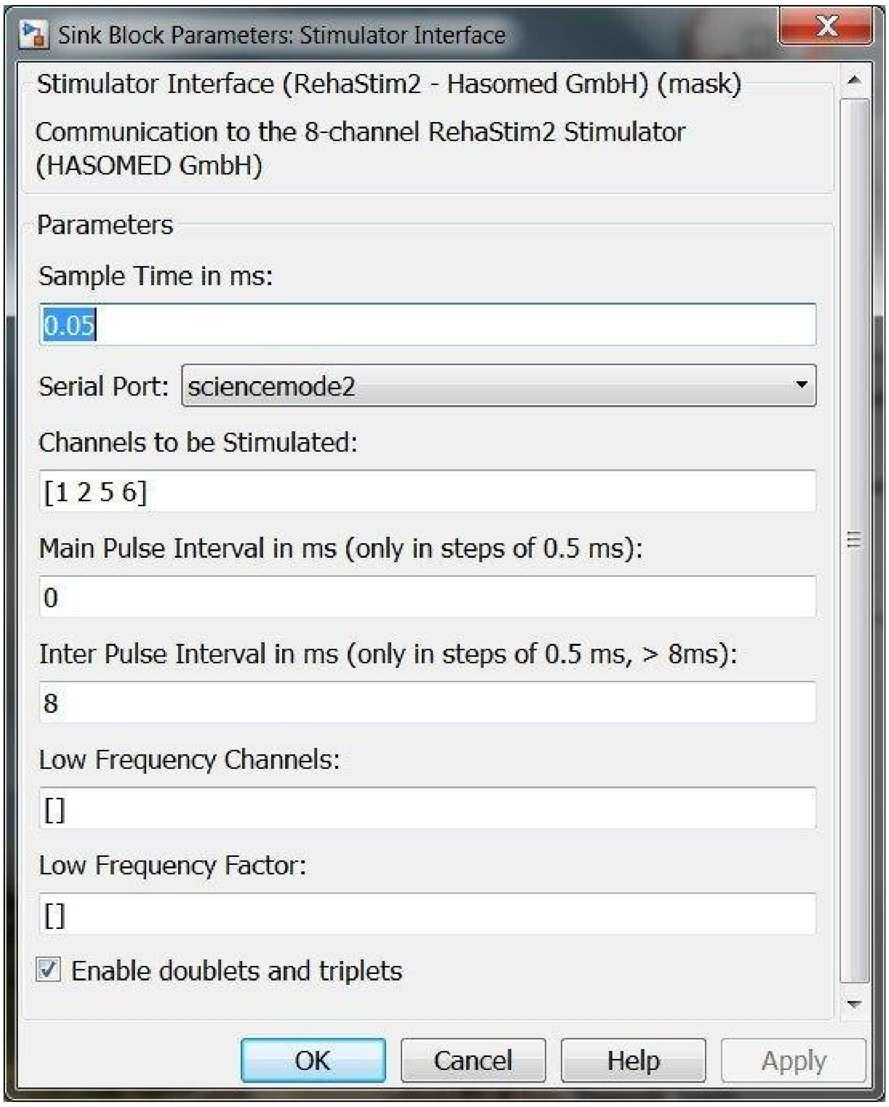
Under **Code Generation** (top level): Choose Language **C**

Under **Code Generation/IDE Link**: Choose Build Option **Build,** choose configuration **Release**

5. Open the library **Sciencemode2_Lib** in Simulink and copy the Stimulator interface block into the Simulink diagram.
6. In the Simulink diagram go to the menu and select under **Simulation/Mode → External.**

4.3 Block parameters

The interface block only supports the **Continuous Channel List Mode (CCLM)** and the **One Shot Channel List Mode (OSCLM)** [1]. For configuration of the stimulator interface some parameters of the stimulator interface block must be specified. The mask of the block is shown below:


Sink Block Parameters: Stimulator Interface

Stimulator Interface (RehaStim2 - Hasomed GmbH) (mask)

Communication to the 8-channel RehaStim2 Stimulator (HASOMED GmbH)

Parameters

Sample Time in ms: 0.05

Serial Port: sciencemode2

Channels to be Stimulated: [1 2 5 6]

Main Pulse Interval in ms (only in steps of 0.5 ms): 0

Inter Pulse Interval in ms (only in steps of 0.5 ms, > 8ms): 8

Low Frequency Channels: []

Low Frequency Factor: []

☑ Enable doublets and triplets

1. The **Sample Time** in milliseconds specifies the update period of the stimulation parameters (not necessary the stimulation period but usually)
2. **Serial Port** specifies the serial port name (e.g. COM1, COM2 … for MS Windows).
3. **Channels to be stimulated:** Defines the active channels in the channel list.
4. **Main Pulse Interval (t_1): Defines the main time, where the stimulation channel list is repeatedly processed. We have $t_1 \geq nPgr \cdot t_2$, where *nPgr* is the maximum of the selected pulse group modes used (nPgr = 1 for single pulses, nPgr = 2 for doublets, nPgr = 3 for triplets). Attention: If t_1 is set to zero then the One Shot Channel List Mode (OSCLM) is activated and the stimulation period is determined by the PC and the specified sample time.**
5. **Inter Pulse Interval (t_2):** Defines the time between two pulse groups, if doublets or triplet are selected. The minimum inter pulse interval is 8 ms per stimulation module

6. **Low frequency channels:** Defines the channels, which have a lower frequency than the normal channels determined by the low frequency factor.
7. **Low frequency factor:** Defines how many times the stimulation is skipped for the channels specified in low frequency channels. (0 =no skip, 1 = skip once etc.)
8. **Enable doublets and triplets:** Must be activated to control the stimulation mode (signal pulse, doublet or triplet) in real-time through an additional input of the stimulator interface.

For more explanations please read the ScienceMode2 – Description and Protocol [1].

4.4 Block inputs

The stimulator interface block processes **two or three inputs** depending on the mask option “Enable doublets and triplets”. All inputs are vectors with the same dimension! This dimension fits to the number of channels specified under “**Channels to be stimulated:”** in the block mask. The inputs are defined as follows:

1. The **first input** is a **vector with all current amplitude values** for the specified channels. Let us assume that the channels 1, 3 and 5 are selected. In this case, the first element of the current amplitude vector is the amplitude of the first channel, the second element is the amplitude of the third channel and the third element is the amplitude of the fifth channel.
2. The second input is a **vector with all pulse width values** for the specified channels.
3. The third optional input is a **vector which specifies the stimulation mode** for the specified channels (0 – single pulse, 1 – doublet, 2 – triplet).

4.5 Building the executable

After a functional diagram has been created, the executable can be build by choosing in the Menu Code/C/C++ Code/Build model or by pressing CTRL-B or by clicking on the icon “Build Model”.

The build process can be monitored in the Matlab Command Window. In case of a successful build the last lines of the output are e.g.:

```
...
### Creating project: C:\Workspace\ScienceStim2_Demo_Windows
### Project creation done.
### Building project...
>>
```

The executable **ScienceStim2_Demo_Windows.exe** is located under **C:\Workspace\ScienceStim2_Demo_Windows\Release** (MS Windows example).

4.6 External Mode in Simulink

The external mode of Simulink can be used to monitor signals and to change parameters online by applying the following steps:

1. Open a command window under MS Windows by starting **cmd.exe.** In Linux open a new terminal / console window.
2. Change to the directory where the executable is located.
3. Start Simulink and open the diagram which was used to build the executable.
4. Start the executable. The stimulation should be active now.
5. In Simulink, choose in the menu **Stimulation/Connect To Target.**
6. Monitor or store signals in Simulink and change parameters if required online.
7. Stop the program in Simulink and select **Disconnect From Target** under Simulation or wait until the specified simulation time has elapsed.

For more information about the Simulink Coder and the External Mode please check the Simulink User Guide and the Simulink Coder User Guide.

5. References

[1] B. Kuberski, *ScienceMode2 – Description and Protocol, Version: 1.2*, HASOMED GmbH, 2012