

Realization of a SIMIT Shared Memory Coupling with Matlab

SIMATIC SIMIT Simulation Platform V9.1

<https://support.industry.siemens.com/cs/ww/en/view/109761656>

Siemens
Industry
Online
Support



Legal information

Use of application examples

Application examples illustrate the solution of automation tasks through an interaction of several components in the form of text, graphics and/or software modules. The application examples are a free service by Siemens AG and/or a subsidiary of Siemens AG ("Siemens"). They are non-binding and make no claim to completeness or functionality regarding configuration and equipment. The application examples merely offer help with typical tasks; they do not constitute customer-specific solutions. You yourself are responsible for the proper and safe operation of the products in accordance with applicable regulations and must also check the function of the respective application example and customize it for your system.

Siemens grants you the non-exclusive, non-sublicensable and non-transferable right to have the application examples used by technically trained personnel. Any change to the application examples is your responsibility. Sharing the application examples with third parties or copying the application examples or excerpts thereof is permitted only in combination with your own products. The application examples are not required to undergo the customary tests and quality inspections of a chargeable product; they may have functional and performance defects as well as errors. It is your responsibility to use them in such a manner that any malfunctions that may occur do not result in property damage or injury to persons.

Disclaimer of liability

Siemens shall not assume any liability, for any legal reason whatsoever, including, without limitation, liability for the usability, availability, completeness and freedom from defects of the application examples as well as for related information, configuration and performance data and any damage caused thereby. This shall not apply in cases of mandatory liability, for example under the German Product Liability Act, or in cases of intent, gross negligence, or culpable loss of life, bodily injury or damage to health, non-compliance with a guarantee, fraudulent non-disclosure of a defect, or culpable breach of material contractual obligations. Claims for damages arising from a breach of material contractual obligations shall however be limited to the foreseeable damage typical of the type of agreement, unless liability arises from intent or gross negligence or is based on loss of life, bodily injury or damage to health. The foregoing provisions do not imply any change in the burden of proof to your detriment. You shall indemnify Siemens against existing or future claims of third parties in this connection except where Siemens is mandatorily liable.

By using the application examples, you acknowledge that Siemens cannot be held liable for any damage beyond the liability provisions described.

Other information

Siemens reserves the right to make changes to the application examples at any time without notice. In case of discrepancies between the suggestions in the application examples and other Siemens publications such as catalogues, the content of the other documentation shall have precedence.

The Siemens terms of use (<https://support.industry.siemens.com>) shall also apply.

Security information

Siemens provides products and solutions with Industrial Security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the Internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place. For additional information on industrial security measures that may be implemented, please visit <https://www.siemens.com/industrialsecurity>.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed at: <https://www.siemens.com/industrialsecurity>.

Table of contents

Legal information	2
1 Introduction	4
1.1 Overview	4
1.2 Principle of operation	5
1.3 Components used	6
2 Engineering	7
2.1 SIMIT Project	7
2.1.1 Project settings	7
2.1.2 Creating SHM coupling in SIMIT	8
2.1.3 Calculation of the size of the SHM	9
2.1.4 Configuration of the chart for data exchange	10
2.2 Shared memory block for MATLAB	11
2.2.1 SHM.cpp	12
2.2.2 Customizing the SHM name	12
2.2.3 Programming input and output connections	13
2.2.4 Programming the SHM functionality	14
2.2.5 Compiling the source code	15
2.3 SIMULINK model	16
2.3.1 Configure simulation parameters	17
2.3.2 Configuring the pause mode	18
2.3.3 Configuring the SHM block	19
2.4 Testing the SHM connection	20
2.5 Synchronization of SIMIT and MATLAB	21
2.5.1 Principle of operation	22
2.5.2 Visual Studio project settings	22
2.5.3 Programming	24
3 Demo project commissioning	25
3.1 Preparation	25
3.2 Starting the application	25
3.3 Starting and stopping of the simulation	27
3.4 Log off client	28
4 Appendix	29
4.1 Service and support	29
4.2 Links and Literature	29
4.3 Change documentation	30

1 Introduction

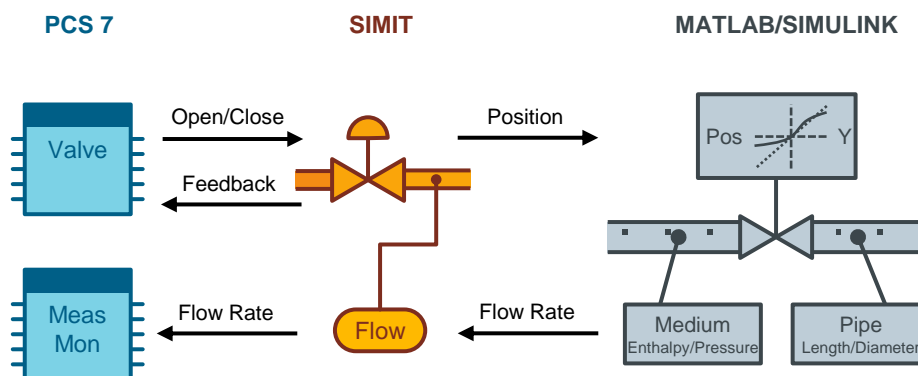
1.1 Overview

The SIMIT Simulation Platform enables comprehensive testing of automation applications and offers a realistic training environment for plant operators even before they are put into operation. In addition, SIMIT enables convenient connection to the SIMATIC PCS 7 process control system with the aid of the couplings to the "VC" virtual controller, to PLCSIM or to the SIMIT unit.

MATLAB is a technical-scientific software for powerful numerical calculations and the professional visualization of data and results. With the SIMULINK extension you design and simulate your system in a model-based manner and have almost limitless possibilities to develop your process model.

Existing SIMULINK process models don't necessarily have to be redeveloped in SIMIT, but can be connected to the SIMIT simulation model via an additional coupling. For example, SIMIT only takes over the simulation of the devices, such as drives and valves ("Device Level"), while in SIMULINK the process ("Process Level") is simulated. Often it suffices to exchange a few values between both systems, for example to calculate the flow in a piping system based on the position of a valve.

Figure 1-1



With SIMIT's Shared Memory Coupling (SHM Coupling), flexible and high-performance interfaces between SIMIT and other applications can be implemented on the same system.

This application example shows you, how to configure a SHM coupling for data transfer between SIMIT and MATLAB and the synchronization of both applications during the simulation.

1.2 Principle of operation

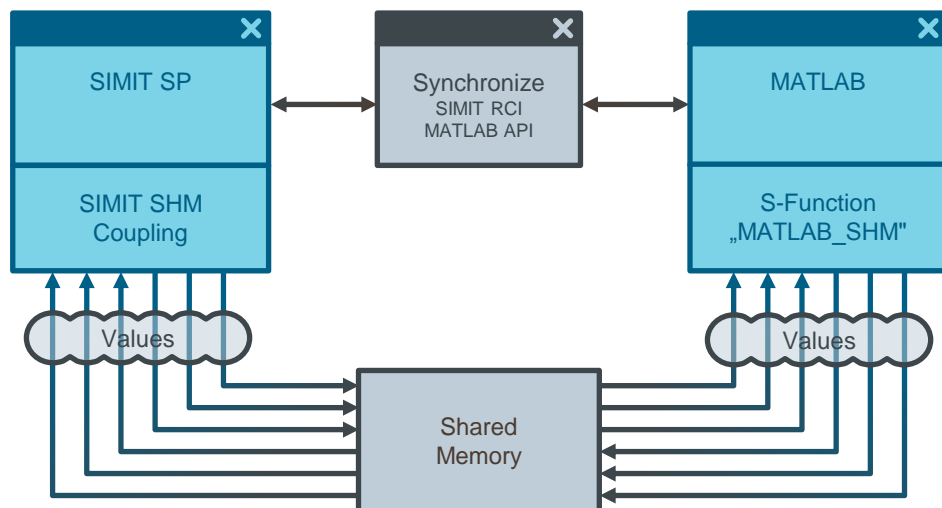
SIMIT and MATLAB simultaneously use a reserved area in the main memory of the host system which is used for data exchange between the two programs.

By default, SIMIT provides a coupling to create or use a shared memory area. MATLAB doesn't have this functionality by default, but it does have a powerful API (Application Programming Interface) that you can use to create your own solutions for using shared memory. For this purpose, a C++ program is compiled to a MATLAB usable system function (S-function).

The synchronization of the SIMIT Simulation Runtime with the solver of the MATLAB/SIMULINK model is assumed by an additional application. This application combines the functions of a SIMIT Remote Control Interface (RCI) client and the MATLAB API. This was created as a Windows Forms application in C#.

The following figure shows the data exchange between SIMIT and MATLAB:

Figure 1-2



1.3 Components used

This application example was created with these hardware and software components:

Table 1-1

Components	Article number	Note
SIMIT SP V9.1 Ultimate	6DL5260-0CX68-0YA5	Shared memory coupling is only included in the SIMIT Ultimate Version.
MATLAB R2018a	-	https://mathworks.com/products/matlab.html
boost C++ libraries	-	www.boost.org
Visual Studio 2017	-	https://visualstudio.microsoft.com
SIMATIC PCS 7 V9.0 SP1	6ES7658-5AX58-0YA5	Connection to PCS 7 is optional.

This application example consists of the following components:

Table 1-2

Components	File name	Note
Documentation	109761656_SIMIT_MATLAB_SharedMem_de.pdf	This document
SIMIT test project MATLAB test project Visual Studio project SyncSiMa application	109761656_SHM_Projects.zip	All necessary projects and files

2 Engineering

2.1 SIMIT Project

Using shared memory coupling (SHM coupling), SIMIT communicates with any other application via a shared memory area on the host system.

For the solution described in this application example, it is a prerequisite that all applications that want to access the common memory area are executed on the same system.

After you start the simulation in SIMIT, SIMIT creates the SHM or opens an existing SHM. For SIMIT to use an existing memory area, it must have the same name and the same size.

The SHM must also be parameterized identically in the other applications that require access to the SHM.

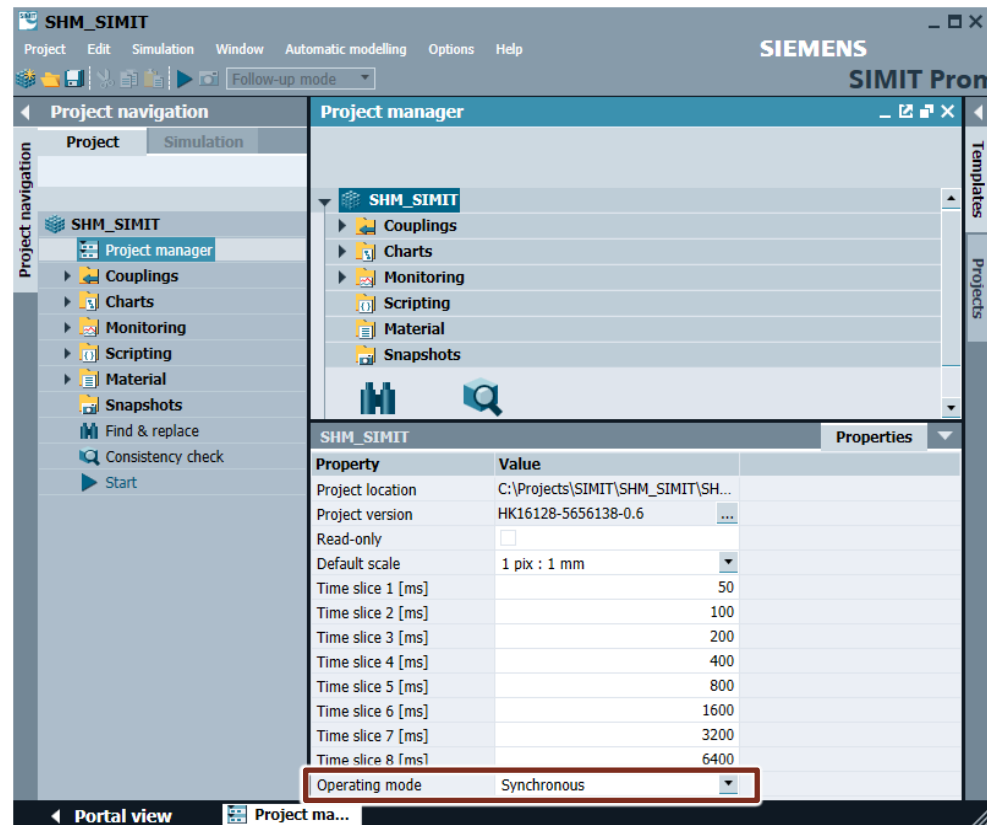
You can download the already completed SIMIT project for this application example from the article page:

<https://support.industry.siemens.com/cs/ww/en/view/109761656>

2.1.1 Project settings

Start SIMIT and create a new project if you have not already done so. Switch to the project view. Open the Project Manager and set the operation mode "Operation mode" to "Synchronous"

Figure 2-1



2.1.2 Creating SHM coupling in SIMIT

To create the SHM link, proceed as follows:

1. In the project navigation, start the function "Couplings > New Coupling".
2. Select the "Shared Memory" link and confirm the selection.
3. Open the newly created "Shared Memory" coupling in the editor.

In the coupling editor, you create the input and output signals and parameterize the SHM. The structure of the SHM consists of the header area and the data area.

The header area is at least 8 bytes in size and has the following structure:

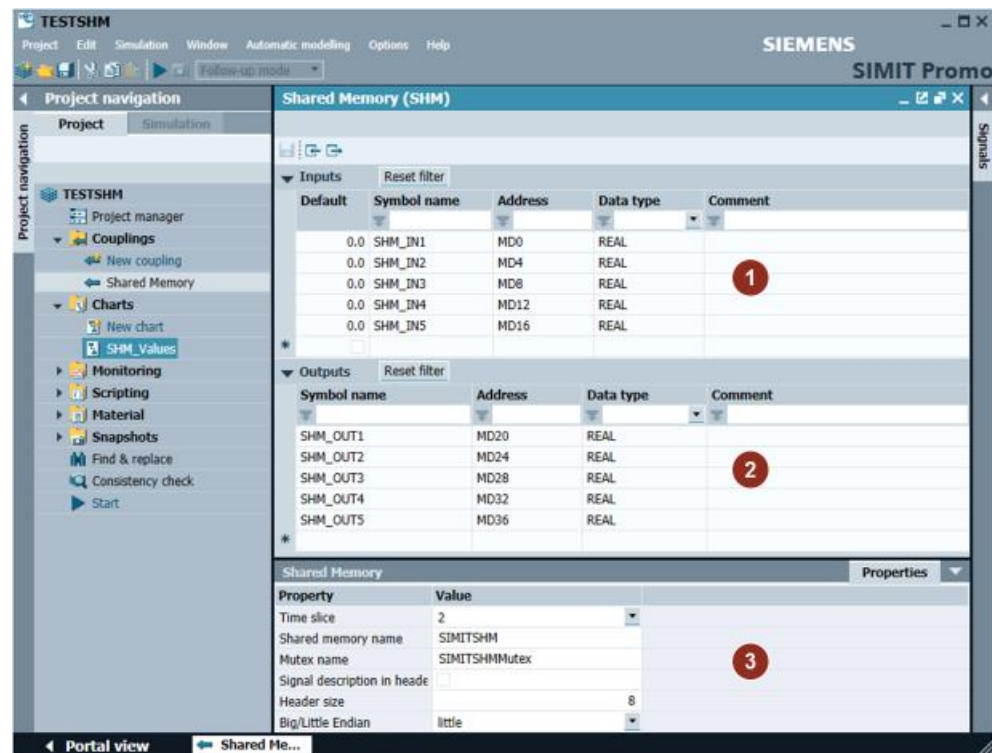
Table 2-1

bytes	Value	Size
0	Size of the memory area	[4 bytes]
1		
2		
3		
4	Size of the header area	[4 bytes]
5		
6		
7		

This example uses the header area with this configuration. Further information on the structure of the SHM can be found in the SIMIT manual:

<https://support.industry.siemens.com/cs/ww/en/view/109750788/73690702987>

Figure 2-2



- (1) Configuration of input signals
- (2) Configuration of output signals
- (3) Parameterizing the SHM

In the shared memory properties you will find some settings, which are briefly explained here:

- Time slice: Cycle in which the SHM is read and written
- Signal description in header: All signals are described in the header. This can be used by other applications to determine the configured signals. The size of the header is calculated and cannot be parameterized.
- Header size: Minimum 8 bytes
- Big/Little Endian: Sets the byte order. Little Endian – The low-order byte is processed first.

2.1.3 Calculation of the size of the SHM

The size of the SHM consists of the header area and the data area. The header size is parameterized in the coupling, while the size of the data area results from the configured signals. The following table shows the size used by the different data types in the SHM:

Table 2-2

SIMIT data types	Allocation in the SHM
BOOL; BYTE	1 byte
WORD, INT	2 bytes
DWORD, DINT, REAL	4 bytes

In the example, 5 real values each are configured as input and output signals. The header size is 8 bytes. Thus, the SHM used is a total of 48 bytes.

$$5 * 4\text{byte} + 5 * 4\text{byte} + 8\text{byte} = 48\text{bytes}$$

Note

SIMIT always reserves a 2048-byte memory area as long as it is not exceeded. If more is required, SIMIT reserves 4096 bytes. For other applications to access the SHM, these must also be configured with 2048 bytes.

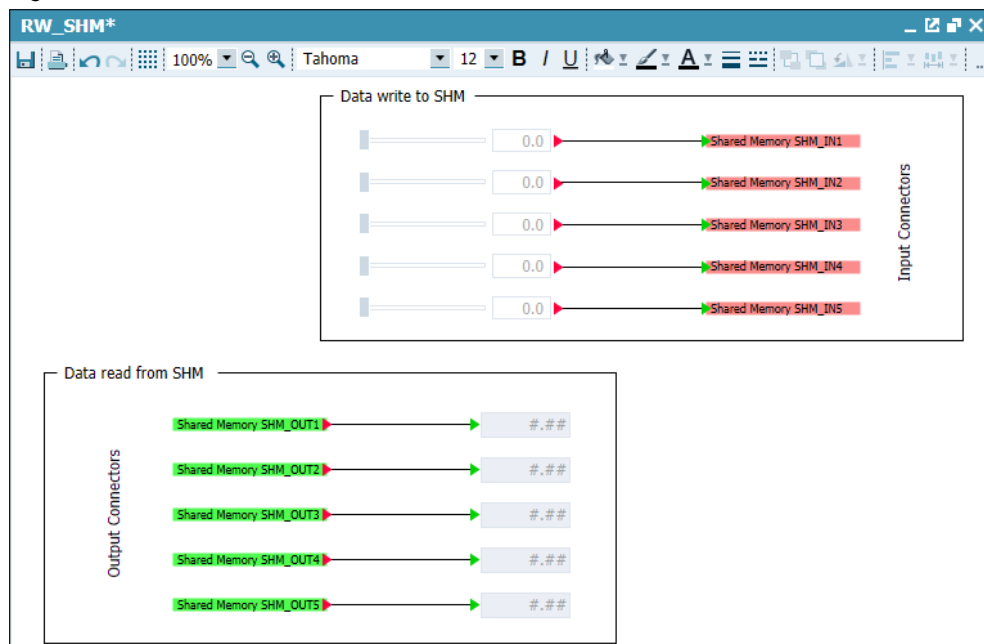
2.1.4 Configuration of the chart for data exchange

Since this example is primarily about the data exchange between SIMIT and the SHM, a complex simulation model is not used. It suffices here to write some values in the SHM and read some values from the SHM.

Proceed as follows:

1. Create a new chart and open it in the editor.
2. Drag the signals from the signal list into the chart while holding down the SHIFT key. In each case five input and output connectors are created.
3. Connect the input connectors to a slider object.
4. Connect the output connectors to a digital display.

Figure 2-3



2.2 Shared memory block for MATLAB

Unlike SIMIT, MATLAB has no built-in SHM interface. However, the powerful MATLAB API allows you to create your own SHM interface. In the following example, an "S-Function" was created with the SHM functionality for MATLAB/SIMULINK. The block was created using the C++ programming language and compiled with the MATLAB Compiler (MEX). For further information regarding MATLAB, please go to:

<https://www.mathworks.com>

Example programs as a template with the SIMULINK framework, such as: For example, "sfun_cppcount_cpp.cpp" can be found in the MATLAB installation directory:

"...\MATLAB\R2018a\toolbox\simulink\simdemos\simfeatures\src"

When programming the SHM functionality, the boost library was used. It contains classes that have already been tested for using Windows shared memory. You can find additional information and the installation files for the boost library here:

<https://www.boost.org>

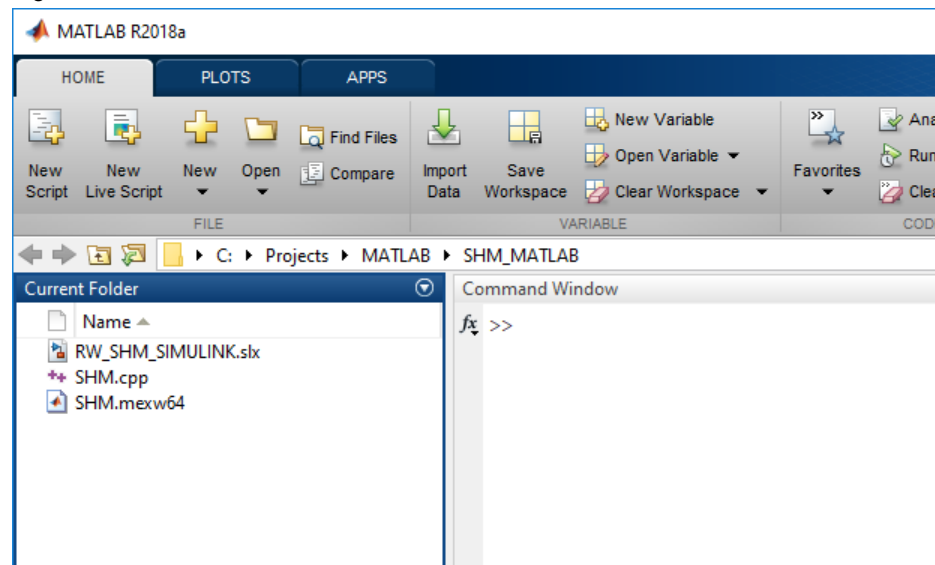
Follow the instructions for installing and precompiling the boost library in "Getting Started on Windows".

You can download the already completed MATLAB/SIMULINK project for this application example from the article page:

<https://support.industry.siemens.com/cs/ww/en/view/109761656>

After extracting the ZIP archive, select this folder in MATLAB as "Current Folder".

Figure 2-4



2.2.1 SHM.cpp

The source code "SHM.cpp" for the SIMULINK block with the SHM function is stored in the project folder of MATLAB "SHM_MATLAB".

The programming contains the basic functionality for SIMULINK blocks. The necessary functions are contained in the header file "simstruc.h".

Under the following link you will find a description of the programming of S functions:

<https://mathworks.com/help/simulink/sfg/example-of-a-basic-c-mex-s-function.html>

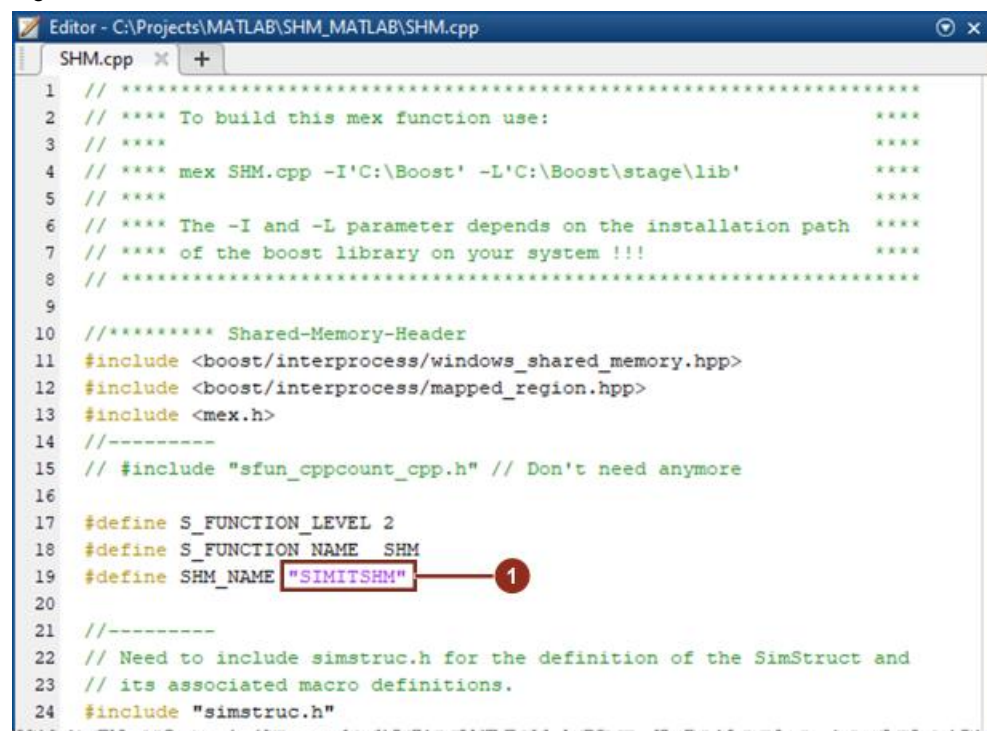
2.2.2 Customizing the SHM name

The name of the SHM in the SIMULINK block must match the name of the SHM in the SIMIT link. In the example, the name is set to "SIMITSHM".

If you want to use a different name for the SHM or another SHM, follow these steps:

1. Open the source file "SHM.cpp" with the editor in MATLAB.
2. Customize the name in the line `#define SHM_NAME "SIMITSHM"` (1).

Figure 2-5

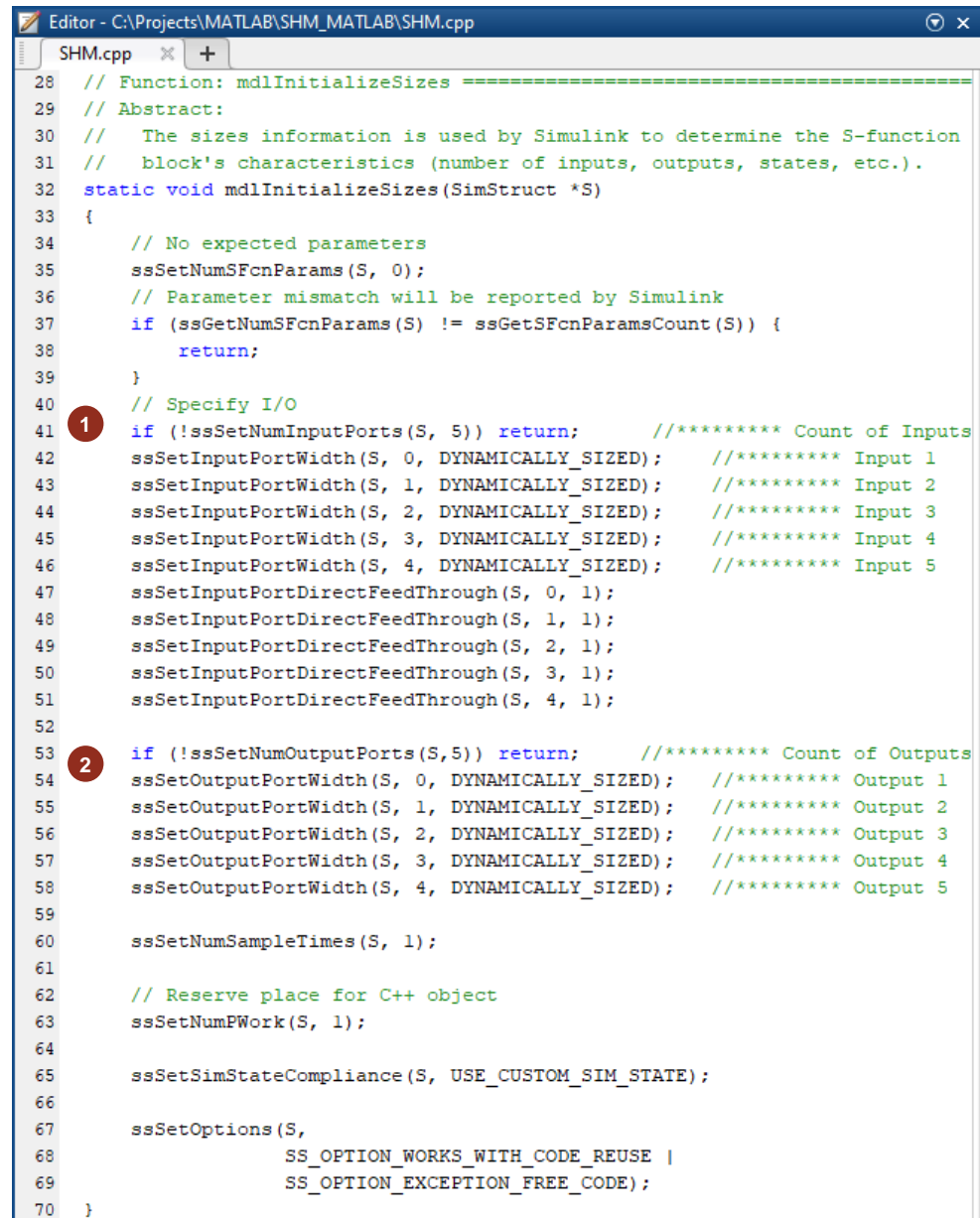


3. Compile the block as shown in chapter [2.2.5 Compiling the source code](#).

2.2.3 Programming input and output connections

Within the function `static void mdlInitializeSizes(SimStruct *S)` the number and size of the block connections are specified. Since the SHM was created in SIMIT with five input and output variables each, you also need five input and output connections here.

Figure 2-6



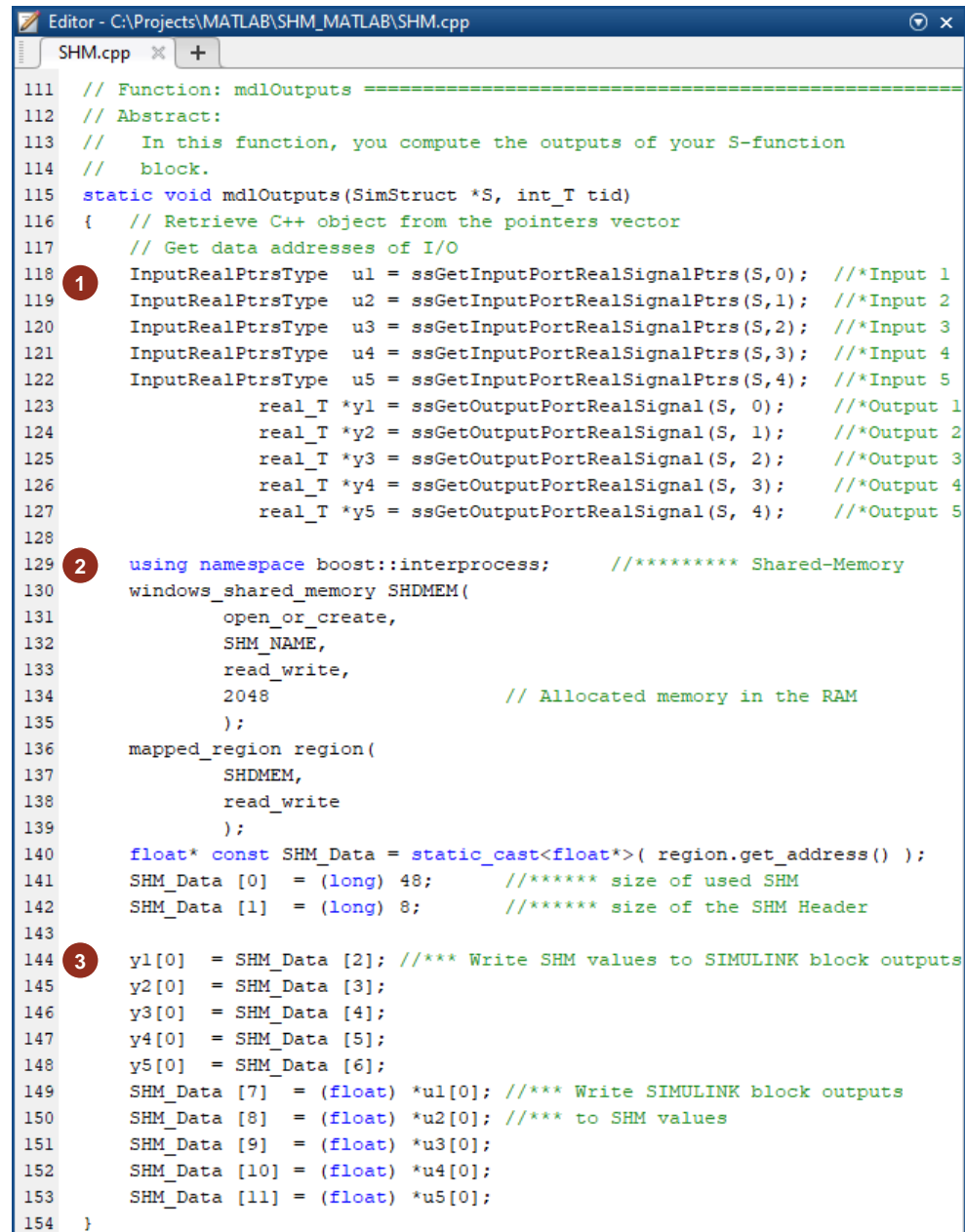
- (1) Number and size of the input connectors
- (2) Number and size of the output connectors

2.2.4 Programming the SHM functionality

The assignment of the internal variables and the call to read and write the SHM takes place in the function:

```
static void mdlOutputs(SimStruct *S, int_T tid)
```

Figure 2-7

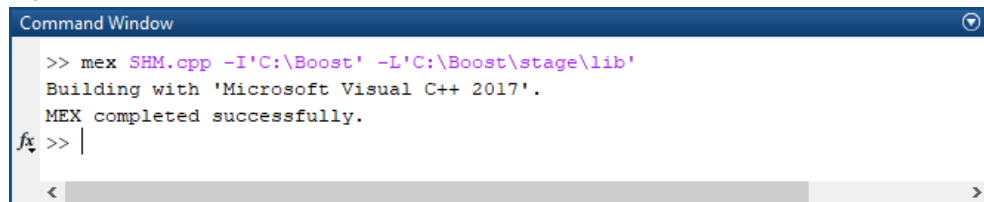


- (1) Reading the input connections of the SIMULINK block
- (2) Calling the shared memory function from the boost library
- (3) Writing the SHM values to the block outputs and the values of the block inputs in the SHM

2.2.5 Compiling the source code

Compile the source code. To do this, in the MATLAB Command Window, use the command: `mex SHM.cpp -I'C:\Boost' -L'C:\Boost\stage\lib'`. After successful compilation the file "SHM.mexw64" will be created in the project folder.

Figure 2-8

A screenshot of the MATLAB Command Window. The title bar says "Command Window". The command prompt shows the command: `>> mex SHM.cpp -I'C:\Boost' -L'C:\Boost\stage\lib'`. The output shows: `Building with 'Microsoft Visual C++ 2017'.` and `MEX completed successfully.`. The prompt is now `>> |`.

```
Command Window
>> mex SHM.cpp -I'C:\Boost' -L'C:\Boost\stage\lib'
Building with 'Microsoft Visual C++ 2017'.
MEX completed successfully.
>> |
```

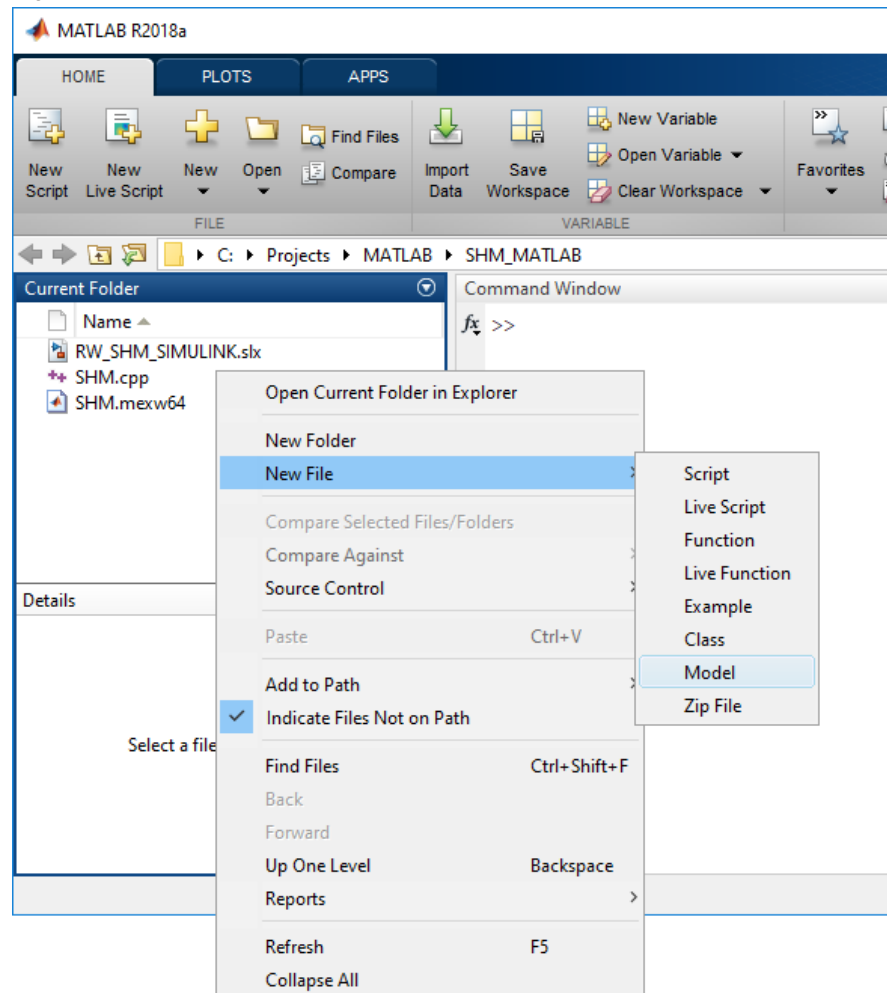
Note

Make sure that the parameters -I and -L point to the correct installation path of the boost library.

2.3 SIMULINK model

To create a new SIMULINK model in MATLAB, select the command "New File > Model" in the context menu of the project folder. This creates a new SIMULINK file with the extension ".slx".

Figure 2-9



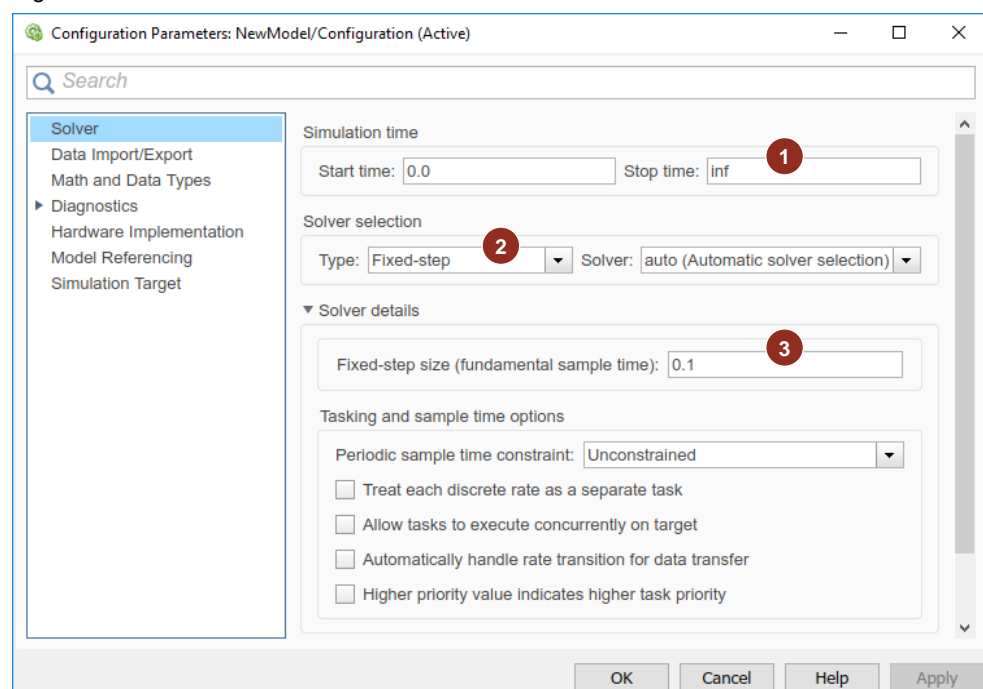
2.3.1 Configure simulation parameters

By default, the MATLAB solver calculates a specific time range and stores the results in a variable array. In order for the values to be exchanged continuously with the SHM, some solver settings are necessary. In addition, the SIMIT and MATLAB simulation steps must be synchronized. For more information, see chapter 2.4.

Proceed as follows:

1. Open the SIMULINK model
2. Open the model settings "Simulation > Model Configuration Parameters"
3. Change the parameters as shown in the following figure:

Figure 2-10



- (1) Stop time inf (infinite) - The calculation runs until you stop it manually.
- (2) Type: fixed step - The calculation steps have a defined size.
- (3) Fixed step size: 0.1 - The calculation steps are set to 0.1 seconds. This corresponds to the time slot with which SIMIT accesses the SHM.

2.3.2 Configuring the pause mode

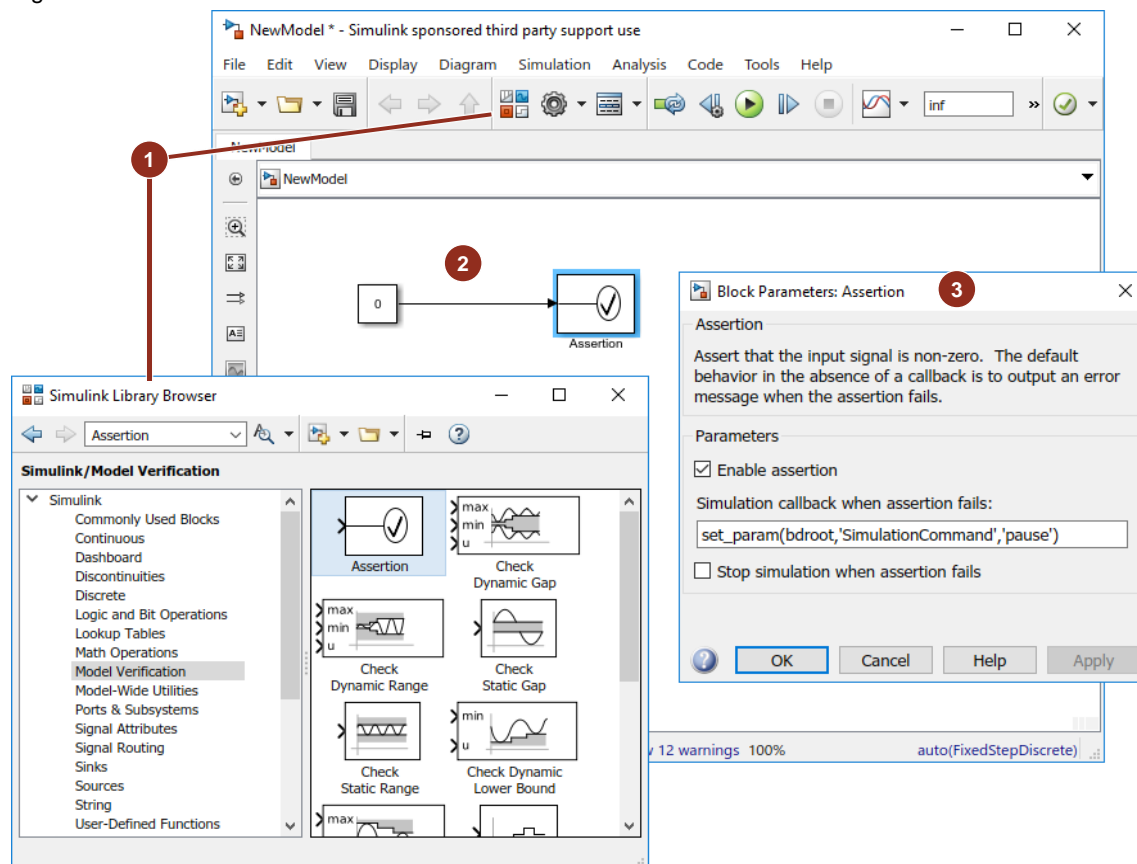
So that MATLAB does not perform the calculation of the simulation permanently and as fast as possible, in this example the calculation is paused after each step. The SIMULINK "Assertion" block enables the execution of a MATLAB statement with which the calculation can be paused. The instruction is executed if the value '0' is present at the input.

Proceed as follows:

1. Open the SIMULINK model and the SIMULINK library browser.
2. Drag and drop the "Assertion" (Simulink > Model Verification) and "Constant" (Simulink > Commonly Used Blocks) into the model.
3. Connect the constant with the value '0' to the assertion block.
4. Open the block parameters of the "assertion" block with a double-click. Enable the Enable assertion option and enter the following MATLAB statement:

```
set_param(bdroot,'SimulationCommand','pause')
```

Figure 2-11



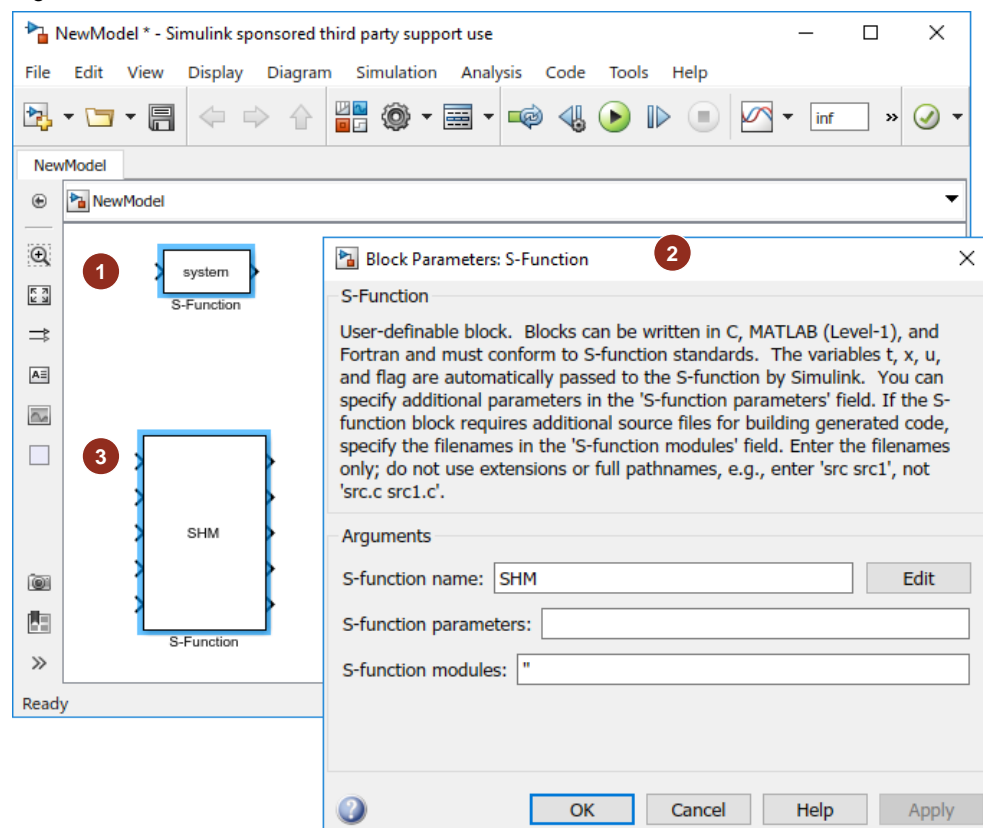
- (1) SIMULINK Library Browser
- (2) "Constant" and "Assertion" blocks
- (3) Assertion block parameters

2.3.3 Configuring the SHM block

Self-programmed blocks are configured in SIMULINK using the "S function". Proceed as follows:

1. Open the SIMULINK model and the SIMULINK library browser.
2. Drag and drop the "S-Function" block (Simulink > User Defined Functions) from the library into the model.
3. Open the parameters of the block with a double-click and enter the name "SHM" of the self-created S function.
4. Adjust the size of the block so that all connections are clearly visible. You can touch the block with the mouse pointer at the corners.

Figure 2-12



- (1) Unconfigured S function block
- (2) S function block parameters
- (3) Configured S function block with the SHM functionality

2.4 Testing the SHM connection

You can now create your SIMULINK model as needed. In the example, the first SIMIT value is read from the SHM and transferred to a rate limiter. The calculated value is returned to the SHM block.

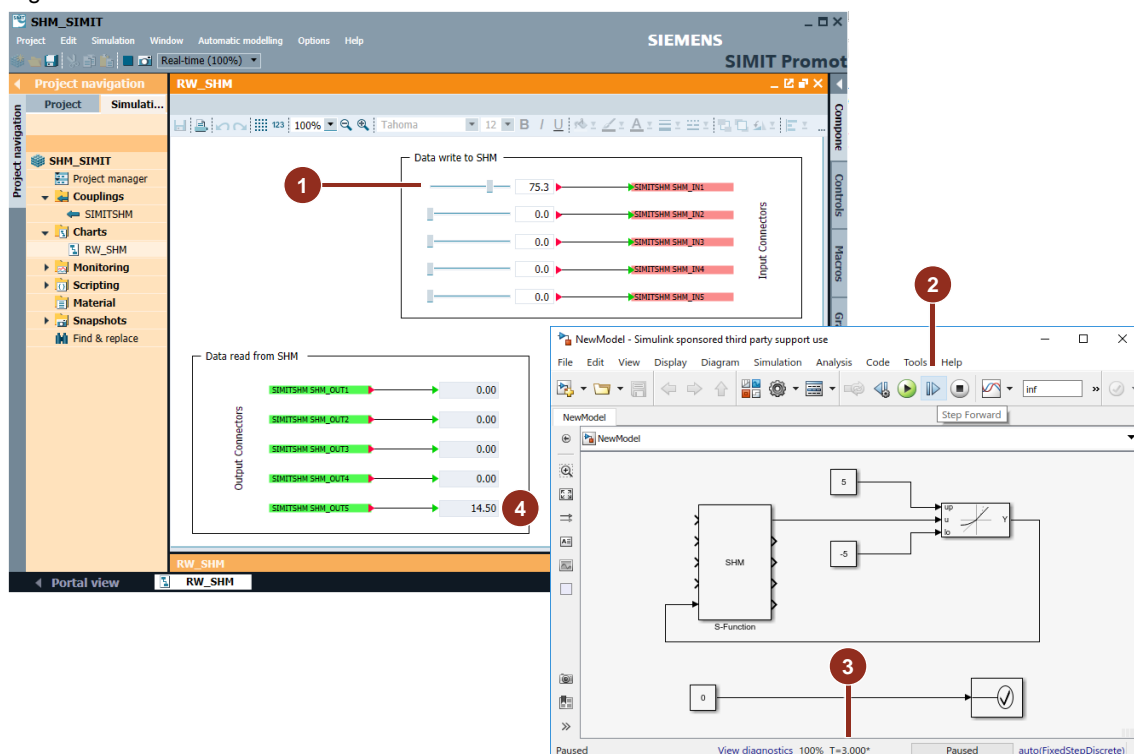
Note

The MATLAB solver only recalculates a block if a value changes at a block input. For the SHM building block, this means that the values from the shared memory are only read out if a value changes at the input.

To test the SHM block, proceed as follows:

1. Open the SIMIT project from chapter: [2.1 SIMIT](#) and start the simulation.
2. Open the SIMULINK model and start the simulation as well.
3. Change the value in SIMIT by means of the slider (1) on the first signal.
4. In MATLAB, continue the calculation step by step with the "Step Forward" button (2). Due to the configured "assertion", the MATLAB simulation is stopped again after each step.
5. In the status bar, you can see the time progress (3) of the simulation.
6. After each step, the value calculated by MATLAB is read in again in SIMIT and displayed (4).

Figure 2-13



2.5 Synchronization of SIMIT and MATLAB

Unlike SIMIT, MATLAB does not calculate the simulation cyclically in a given time interval, but by default performs the calculation from start to finish as fast as possible and stores all the results in a data field.

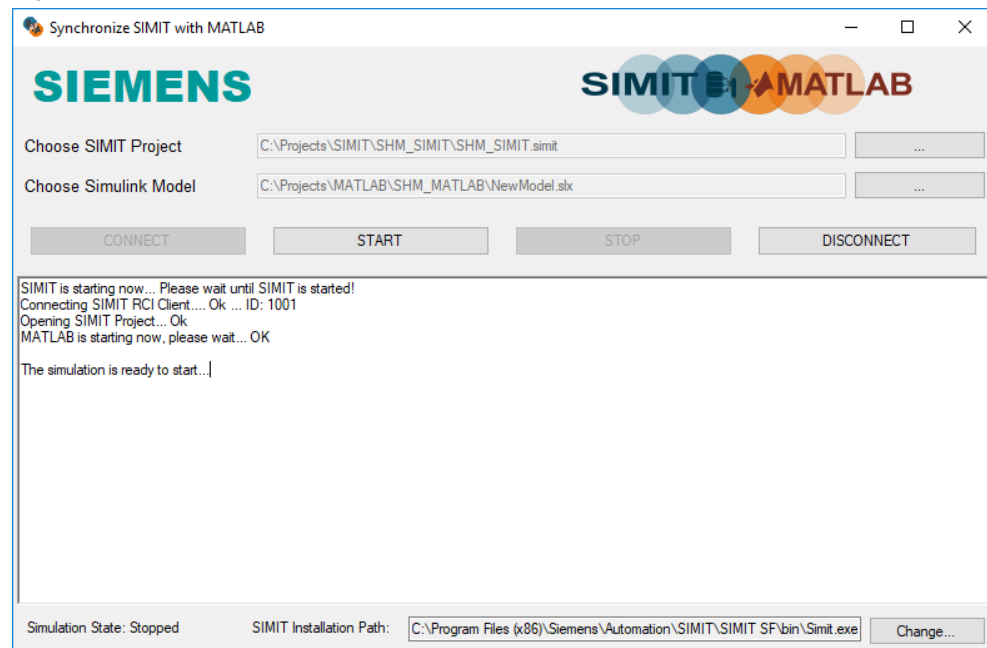
Since MATLAB goes into pause mode after each calculation step due to the configured "assertion" function, MATLAB must now be made to calculate the next step with the aid of an external application.

For this purpose, a Windows-Forms application was created in the C# programming language. The application can connect to SIMIT as a WCF client and access the MATLAB API. The application must be started on the system where SIMIT and MATLAB are installed.

You can download the Windows Forms application "SyncSiMa" and the associated Visual Studio project from the entry page:

<https://support.industry.siemens.com/cs/ww/en/view/109761656>

Figure 2-14



Note

This application is an example program. There can be no assurance that it will run under all conditions or that the SIMIT and MATLAB applications will respond in the manner described.

In the event that you wish to make changes to the application or to extend the application, the source code will be in the form of a Visual Studio 2017 project.

2.5.1 Principle of operation

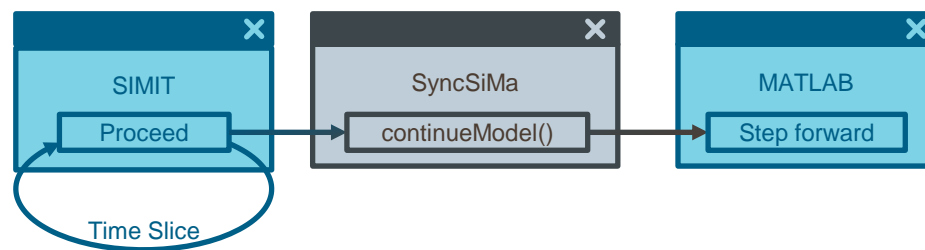
After starting "SyncSiMa", you must select a SIMIT and a MATLAB project. Then you can connect the application as a SIMIT client.

Note

The application only works if the SIMIT project is operated in synchronized mode.

SIMIT generates a "Proceed" call in synchronized mode after every run for all registered clients. The program "SyncSiMa" receives this call and then issues the command "SimulationCommand: Continue" to MATLAB. After processing MATLAB returns to pause mode and waits for the next "continue" command.

Figure 2-15



For more information on the SIMIT operating modes, refer to the "SIMIT - Remote Control Interface (RCI)" manual at the following link:

<https://support.industry.siemens.com/cs/ww/en/view/93763144/70169049483>

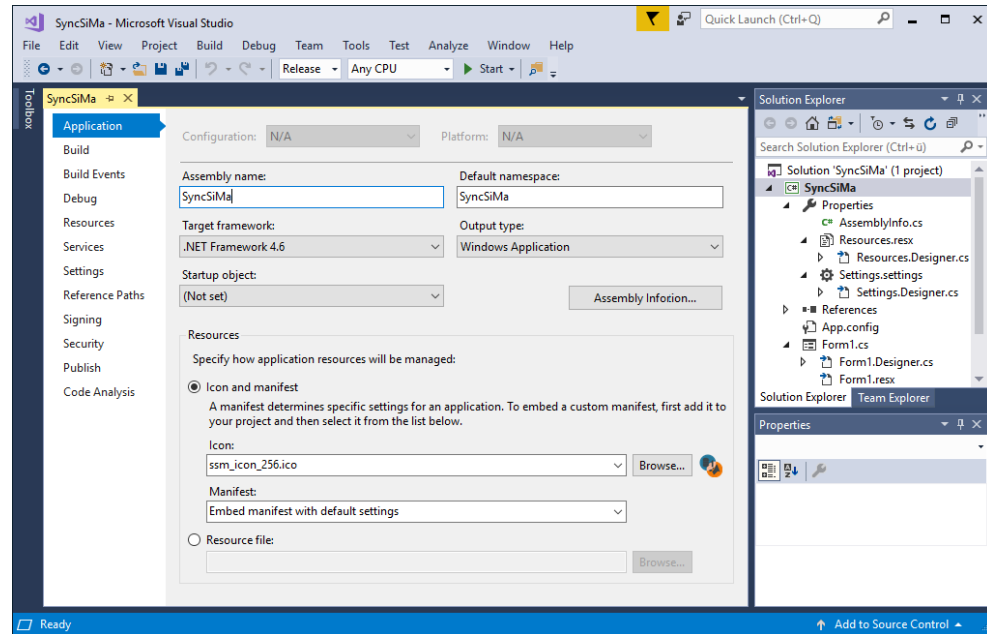
2.5.2 Visual Studio project settings

The program was created with Visual Studio 2017 as C# Windows Forms application. If necessary, you can open the supplied project and customize the programming to your own requirements. The project is enclosed with the application example under the name "SyncSiMa". The following settings and references are required to use the SIMIT Remote Control Interface (RCI) and the MATLAB Application Programming Interface (API).

Dot NET version

SIMIT provides the ".NET Framework V4.6" with the installation. For this reason, the application was also created with version 4.6.

Figure 2-16

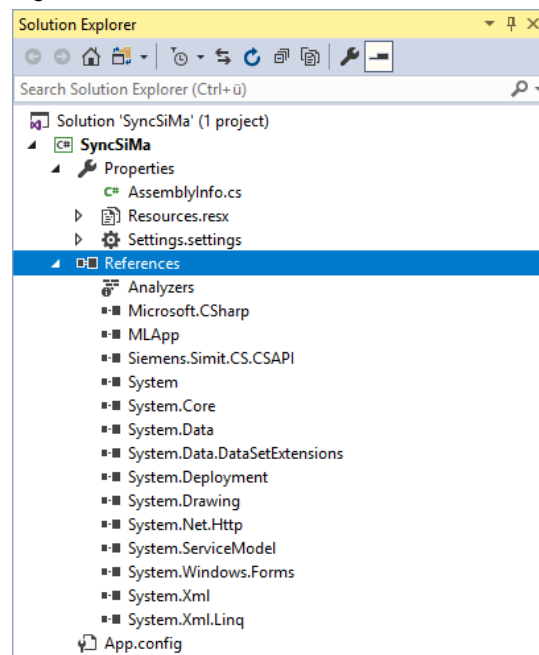


References

The following additional references are required for successful compilation:

- MLApp
"Add Reference > COM > Matlab Application (Version 9.4) Type Library"
- Siemens.Simit.CS.CSAPI.dll

Figure 2-17



2.5.3 Programming

The basis for the programming of this application is the SIMIT documentation for using the RCI interface on the one hand and the documentation for the MATLAB API on the other.

Manual: SIMIT – Remote Control Interface (RCI):

<https://support.industry.siemens.com/cs/ww/en/view/93763144>

MATLAB documentation:

<https://mathworks.com/help/matlab/programming-interfaces-for-external-languages.html>

Note

This manual does not specifically discuss how to program the application. Basic knowledge of programming with C# is required.

The class "SiMaConn" is derived from the class "Client" from the example "Implementation of a passive, synchronized client" in the manual:

<https://support.industry.siemens.com/cs/ww/en/view/93763144/70784194699>

In addition, it has been enhanced with features for controlling MATLAB.

The following figure shows the programming of the "Proceed" call to continue the MATLAB simulation after each cycle of SIMIT.

Figure 2-18

```
using System.Threading;
using Siemens.Simit.CS.CSAPI;

namespace SyncSiMa
{
    public class SiMaConn : IControlSystemCallback
    {
        public static IControlSystem proxy;
        public static ControlSystemServiceParams parameters;
        public static ControlSystemResult result;

        public void SimCommandCanExecute(ControlSystemServiceParams parameters) ...

        public void SimCommandDoExecute(ControlSystemServiceParams parameters)
        {
            // Console.WriteLine("SimCommandDoExecute received, Service=" + parameters.Service);
            switch (parameters.Service)
            {
                case ControlSystemService.Proceed:
                    // Console.WriteLine("--> Proceed: " + parameters.IntVal1 + "ms ");
                    commandReceivedParameters = parameters;
                    new Thread(new ThreadStart(doCommand)).Start();
                    break;
                case ControlSystemService.Terminate:
                    isConnected = false;
                    break;
            }
        }

        public void SimCommandAnswerDoExecute(clientID, ControlSystemResult result, string ...

        private void doCommand()
        {
            // proceed command from SIMIT
            matlab.Execute(string.Format("cd {0}", _matpath));
            matlab.Execute(string.Format("set_param('{0}','SimulationCommand','continue')", _matmodel));
            proxy.SimCommandAnswerDoExecute(clientID, ControlSystemResult.Ok, null, commandReceivedParameters.Service);
        }
    }
}
```


3 Demo project commissioning

The following licensed software products are required for the operation of the enclosed projects:

- SIMIT Simulation Platform V9.1
- MATLAB R2018a

If you have not already done so, download the demo projects from the Articles page of the sample application:

<https://support.industry.siemens.com/cs/ww/en/view/109761656>

3.1 Preparation

Extract the ZIP archive "109761656_SHM_Projects.zip". It contains the following other archives:

- SHM_SIMIT.simarc
- SHM_MATLAB.zip
- SHM_SyncSiMa_Application.zip
- SHM_SyncSiMa_VSProject.zip

Extract the ZIP archives "SHM_MATLAB.zip" and "SHM_SyncSiMa_Application.zip". Dearchive the SIMIT archive with the SIMIT function "Dearchive". Then close SIMIT again.

The application "SyncSiMa" is created so that it can be used with any SIMIT project and any SIMULINK model.

You only need the Visual Studio project "SHM_SyncSiMa_VSProject.zip" if you want to make changes to the "SyncSiMa" program.

3.2 Starting the application

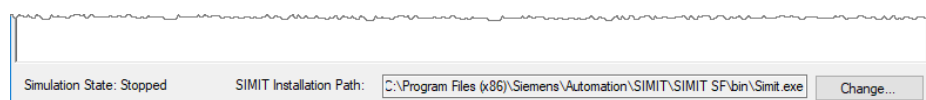
Proceed as follows:

1. First start SIMIT without loading a project.

Note

Although the "SyncSiMa" application has the option of detecting whether SIMIT has already been started or not, problems with the SIMIT license server may occur when it is first started. SIMIT is then started in demo mode, in which the projects offered do not work.

2. Start the "SyncSiMa" application with a double-click. At the same time, an instance of MATLAB is opened in the background. This can take a moment.
3. If you have not installed SIMIT in the standard path shown below, first select the current SIMIT installation path.

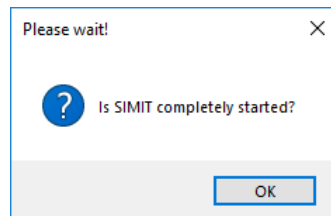


3 Demo project commissioning

4. Select the previously extracted SIMIT project and the SIMULINK model using the application's object browser. The "CONNECT" button becomes active.

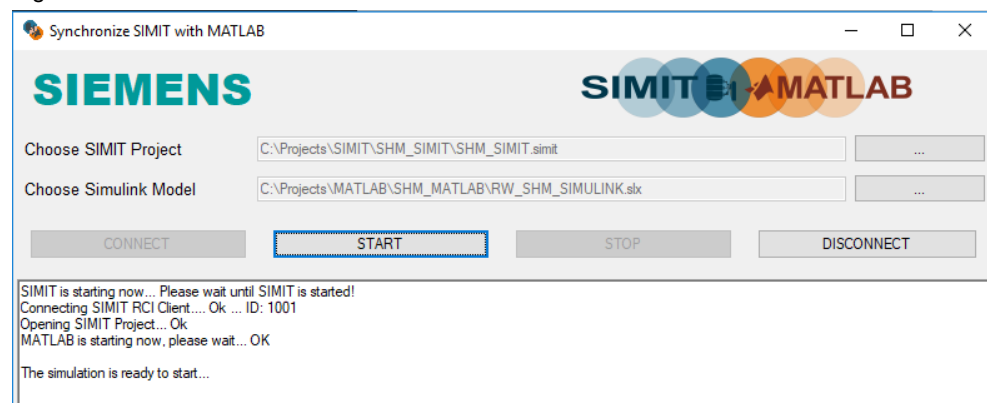


5. Press the "CONNECT" button. The following functions are performed by it:
 - Starting SIMIT if not already done. If SIMIT has to be started, wait until SIMIT is completely started and then confirm the dialog.



- The application logs on to the SIMIT server as a client.
- The selected SIMIT project opens.
- The selected SIMULINK model is loaded.
- The "START" and "DISCONNECT" buttons become active.

Figure 3-1



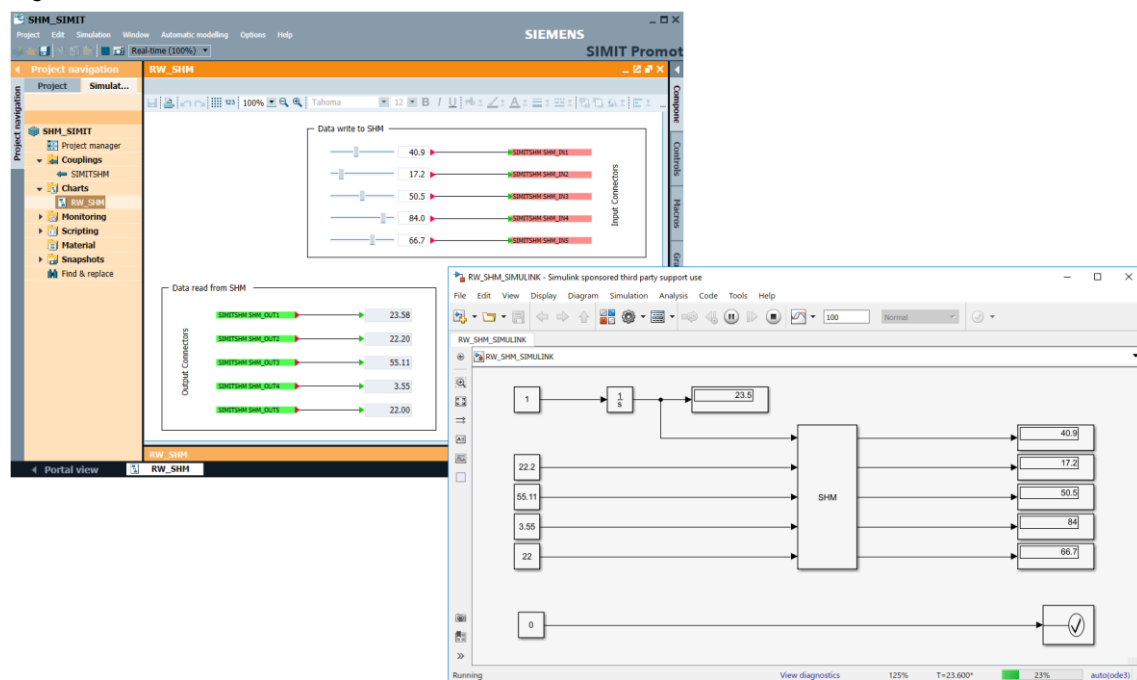
3.3 Starting and stopping of the simulation

The simulation can now be started in synchronized operation. To do this, click the "Start" button. After the successful start of the two simulations, the "STOP" button becomes active.

You now have the option to switch between the applications to observe and control the simulation. To do this, open the chart "RW_SHM" in the SIMIT project view.

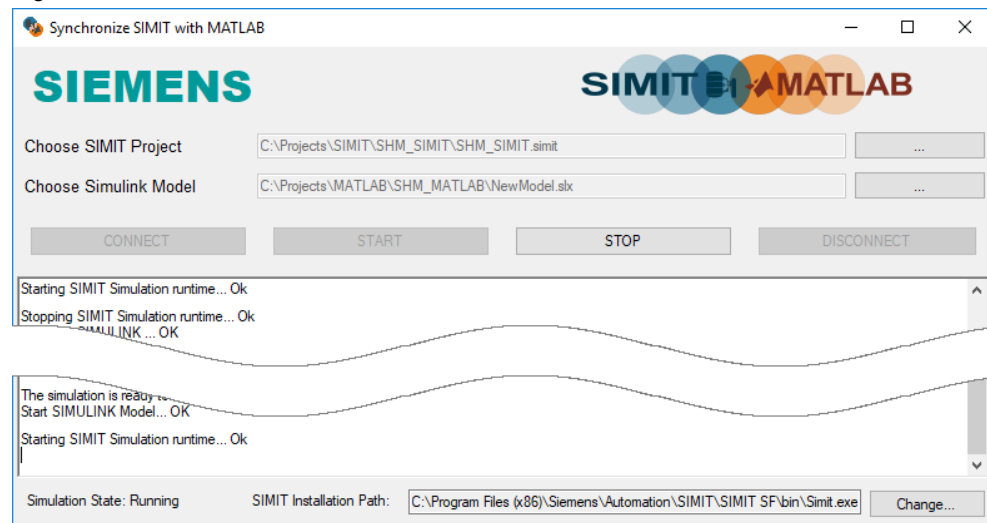
By changing the values in SIMIT or the constants in SIMULINK, you can follow the changes in the other program.

Figure 3-2



After the simulation you return to the "SyncSiMa" application and click on the "STOP" button

Figure 3-3

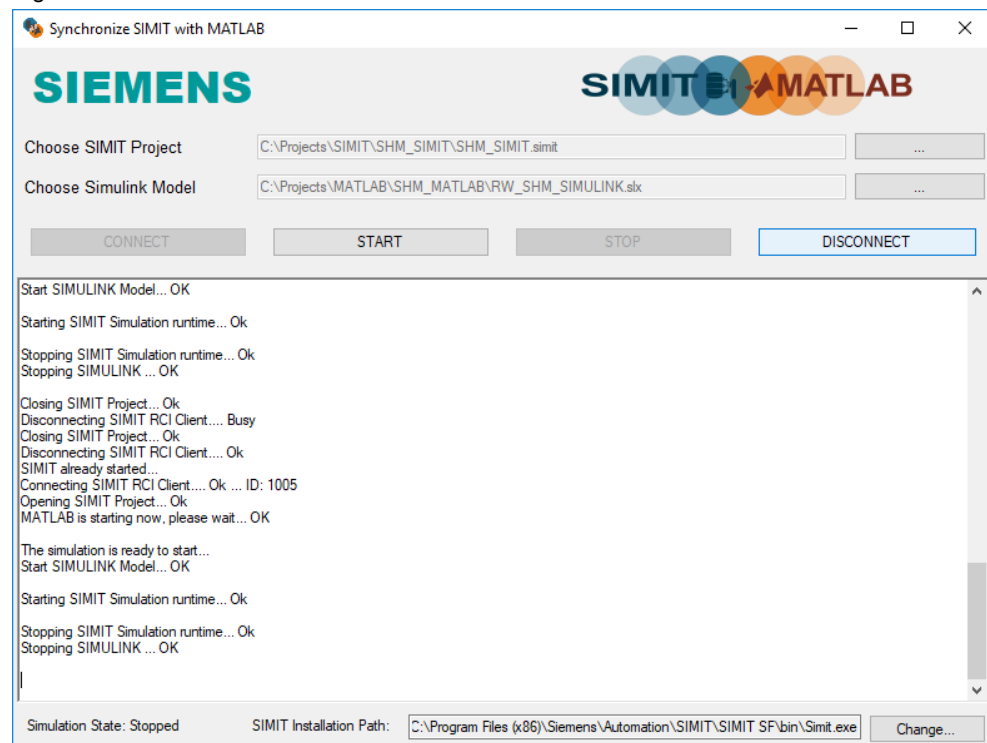


3.4 Log off client

The SIMIT project is closed using the "DISCONNECT" button and the client is logged off.

You can now close the application or select new projects and reconnect the client for another simulation.

Figure 3-4



4 Appendix

4.1 Service and support

Industry Online Support

Do you have any questions or need assistance?

Siemens Industry Online Support offers round the clock access to our entire service and support know-how and portfolio.

The Industry Online Support is the central address for information about our products, solutions and services.

Product information, manuals, downloads, FAQs, application examples and videos – all information is accessible with just a few mouse clicks:

support.industry.siemens.com

Technical Support

The Technical Support of Siemens Industry provides you fast and competent support regarding all technical queries with numerous tailor-made offers – ranging from basic support to individual support contracts. Please send queries to Technical Support via Web form:

www.siemens.com/industry/supportrequest

SITRAIN – Training for Industry

We support you with our globally available training courses for industry with practical experience, innovative learning methods and a concept that's tailored to the customer's specific needs.

For more information on our offered trainings and courses, as well as their locations and dates, refer to our web page:

www.siemens.com/sitrain

Service offer

Our range of services includes the following:

- Plant data services
- Spare parts services
- Repair services
- On-site and maintenance services
- Retrofitting and modernization services
- Service programs and contracts

You can find detailed information on our range of services in the service catalog web page:

support.industry.siemens.com/cs/sc

Industry Online Support app

You will receive optimum support wherever you are with the "Siemens Industry Online Support" app. The app is available for Apple iOS, Android and Windows Phone:

support.industry.siemens.com/cs/ww/en/sc/2067

4.2 Links and Literature

Table 4-1

No.	Topic
\1\	Siemens Industry Online Support https://support.industry.siemens.com
\2\	Link to the article page of the application example https://support.industry.siemens.com/cs/ww/en/view/109761656
\3\	Manual - SIMATIC SIMIT Simulation Platform (V9.1) https://support.industry.siemens.com/cs/ww/en/view/109750788
\4\	Manual - SIMIT Remote Control Interface (RCI): https://support.industry.siemens.com/cs/ww/en/view/93763144
\5\	The Mathworks – MATLAB and SIMULINK https://mathworks.com
\6\	Boost library https://www.boost.org

4.3 Change documentation

Table 4-2

Version	Date	Change
V0.9	10/2018	Review edition