Reference manual
20-sim 4C 2.1

Windows XP / Vista / 7 / 8

20-sim 4C 2.1 Reference Manual

Author: Ir. C. Kleijn,

Disclaimer

This manual describes the prototyping environment 20-sim 4C.

Controllab Products B.V. makes every effort to insure this information is accurate and reliable. Controllab Products B.V. will not accept any responsibility for damage that may arise from using this manual or information, either correct or incorrect, contained in this manual.

Information in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Controllab Products B.V.

Windows is a registered trademark of the Microsoft Corporation, USA.

Reference

Kleijn, C., Differ H.G.
20-sim 4C 2.1 Reference Manual
Enschede, Controllab Products B.V., 2013
ISBN 978-90-79499-14-4

Information

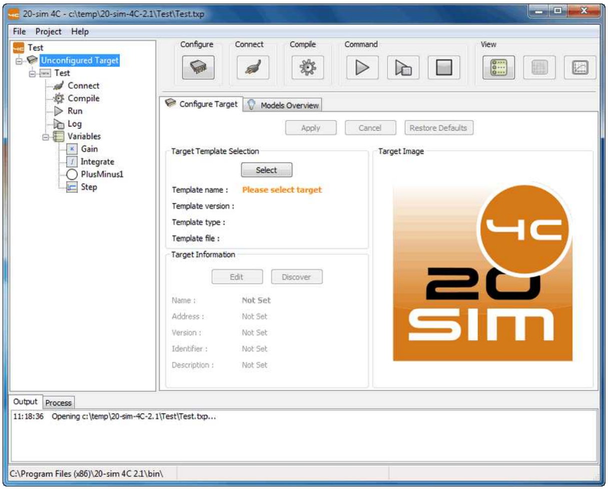
Controllab Products B.V.
Address: Hengelosestraat 500
7521 AN Enschede
the Netherlands
Phone: +31-85-773 18 72
Internet: www.20sim4C.com
www.20sim.com
www.controllab.nl
E-mail: info@20sim.com

Table of Contents

1 Introduction

20-sim 4C is a prototyping environment that enables you run C-code on on hardware like PC's or ARM-9 based processor boards. The C-code may be hand-written or automatically generated. This enables you to perform various tasks:

- **Measurement and Calibration**: From 20-sim 4C you can export C-code that will operate and read sensors.
- **Machine Control**: With 20-sim 4C you can export code to external targets to control the operation of machines. In 20-sim 4C you can start and stop the controller and change parameters during run-time.
- **Rapid Prototyping**: By generating C-code automatically from external programs like 20-sim, a rapid cycle of designing (20-sim), generating code (20-sim) and testing (20-sim 4C) can be performed.



20-sim 4C - c:\temp\20-sim-4C-2.1\Test\Test.txp

Template name : Please select target
Template version :
Template type :
Template file :

Name : Not Set
Address : Not Set
Version : Not Set
Identifier : Not Set
Description : Not Set

11:18:36 Opening c:\temp\20-sim-4C-2.1\Test\Test.txp...

C:\Program Files (x86)\20-sim 4C 2.1\bin\

20-sim 4C helps you to get 20-sim models running as C-code on real-time processors.

Key Features

- **Sources**: 20-sim 4C will accept handwritten C-code and automatically generated C-code from external programs like 20-sim.
- **Targets**: With 20-sim 4C you can run code on any target that runs a RTAI Linux operating system like PC's, PC/104 boards and ARM processors.
- **20-sim**: Any 20-sim model can be exported to C-code and imported in 20-sim 4C.
- **Command**: From 20-sim 4C you can start and stop running code on a target processor. During run-time you can change parameter values.
- **Monitoring and Logging**: In 20-sim 4C you can monitor and log every variable that is available on the target. Values are transmitted from the target to 20-sim 4C in real-time with sample rates up-to 100 Hz or off-line with the full sample rates up to 50 kHz.
- **Communication**: 20-sim 4C runs on a Windows PC. Communication between 20-sim 4C and a target is performed through ethernet using the TCP/IP protocol.
- **Speed**: Sample rates up to 50 kHz can be achieved, depending on the type of processor and the used I/O hardware.

2 What is new in 20-sim 4C

New Features

1. Support for xenomai targets.

2. Check for updates when a newer version of 20-sim 4C is available.

3. Improved Code-Engine :
   Target can be changed (e.g. swith from Bachmann to TS-arm) without regeneration of code from 20-sim.
   The code templates and the source code framework have been updated which allows multiple code creators.

4. Experimental support for Matlab as code creator. (works on all linux/rtai/xenomai targets.)

5. Experimental support for Scilab as code creator. (works on all linux/rtai/xenomai targets.)

6. Optional finish time, run the application forever.

7. Context help for each target from the Target Information dialog. Help shows configuration options and pin layout.

8. Present a small icon for each target in the target overview

9. New Controllab house style for icons and dialogs.

10. Unattended installation. (usage 20-sim-4C-2.1.X.X-win32.exe /S)

General Improvements

1. Improved the TCF format for initialization and deinitialization for I/O components.

2. Updated various targets, see target/<target name>/doc/Changelog.txt for details per target.

3. Updated license dialog, improved support for updating licenses and floating license support.

4. Added scrollbars to main window for small screens.

5. Removed double scrollbars for tree controls when resizing to small.

6. C-Code: added "custom_pre_io_func" and "custom_post_io_func" functions for custom I/O initialization and termination.

7. C-Code: warn about unimplemented SIDOPS functions

8. Installer: removed the dos-box during installation.

Bug fixes

1. Layout of the TargetOverview window when switching target.

2. License: error where 20-sim 4C could not go to demonstration mode is fixed.

3. License: minor fix when trying to activate a license that does not exist.

4. License: activation now checks if "All Users" is possible. If not this option is greyed out.

5. C-Code: Initialize hidden parameters and hidden initial values.

6. C-Code: Replace unknown 20-sim token XX_NR_INITIALVALUE_FUNCS by NUMBEROF_INITIALFUNCTION

7. C-Code: Fix the implementation of the dly() function

8. Fixed: crash on unloading a monitor window with open 3D animation on systems with older OpenGL drivers

TS-7300

1. Two TS-7300 target configurations are available:
   (1) TS-7300 with default FPGA configuration 2xPWM and 2xEncoder
   (2) TS-7300 with CUSTOM FPGA configuration that enables 5xPWM and 5xEncoder
   Both configurations have support for the TS-9700 and TS-ADC16 I/O boards.

Bachmann

1. Support for logging.

2. Card discovery is rewritten, real, virtual, network mapped modules will be found. No need to list in the mconfig.ini

3. Store parameters & variables in xml format on target.
   All settings/parameters/values are retrieved from model configuration file. (Will be stored on target)

4. Bachmann show busy percentage (CPU usage) in the model browser.

5. No need to regenerate code with 20-sim for e.g. time base or changed parameters.

6. Check and adapt time base if possible.
   When a module is started a check is performed if the sample time is an exact multiple of the sync time.
   If so a factor is calculated and set. If no exact multiple can be found the module will not be started.

7. Multiple 20-sim modules can be ran at the same time.
   Module name is the generated submodel name. (xxsim module name is no longer used)
   (restriction module names may not exceed 8 characters.)

8. Setting the channel mode is now performed from 20-sim 4C, no need to the configure the mconfig.ini manually.

9. On reboot the connection will be restored. (no need to close & open 20-sim 4C anymore)

10. Updated initialization and deinitialiazation of all supported cards.
    Make sure you update your own configuration accordingly to make sure all I/O components are properly initialized and deinitialized.

11. Updated documentation for Bachmann Target usage e.g. for configuration with 20-sim and setting the time base.

3 Contact

Please contact Controllab Products if you want to know more about 20-sim 4C or the supported hardware.

Controllab Products B.V.
www.20sim4C.com
www.20sim.com
www.controllab.nl

4 Installation

4.1 Requirements

20-sim 4C only works on computers that will meet the following requirements:

- Operating System: Windows XP, Vista, 7 or 8.
- Available Disk Space: 350 MB.
- Ethernet: an ethernet connection is required for external targets.

Further requirements for using 20-sim 4C:

- 20-sim 4.1.2.4 Professional (or higher) installed on the same computer.
- A valid 20-sim 4C license. Contact your local distributor to obtain a valid license. Note that 20-sim 4.1 (or higher) Professional and the 20-sim 4C 2.1 have separate licenses!

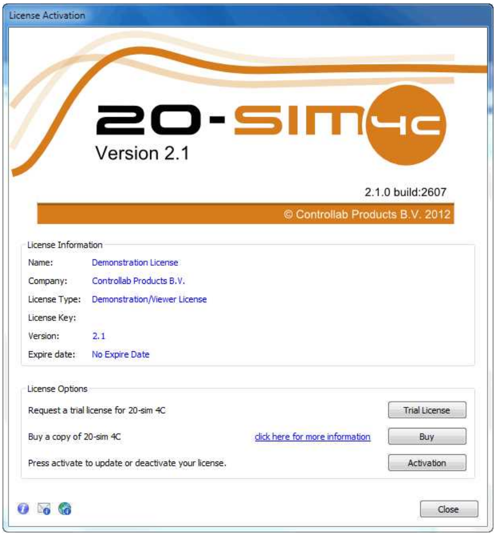
4.2 Installing 20-sim 4C

20-sim4C can be downloaded from the website www.20sim4C.com. This is an installation file that will install 20-sim on your computer. In this topic we describe how to obtain a valid license for 20-sim.

1. **Install 20-sim 4C** on the same computer as 20-sim 4.1 Professional or higher.

2. From the Windows **Start menu** open **20-sim 4C**.

If a valid license of 20-sim 4C was activated before, the program will start automatically. If you have not installed 20-sim 4C before, the *License Activation* dialog will open:



License Information

Name: Demonstration License

Company: Controllab Products B.V.

License Type: Demonstration/Viewer License

License Key:

Version: 2.1

Expire date: No Expire Date

License Options

Request a trial license for 20-sim 4C

Buy a copy of 20-sim 4C click here for more information

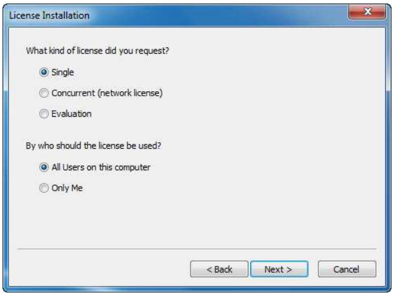
Press activate to update or deactivate your license.

20-sim 4C 2.1 License Activation Dialog.

3. If you have a valid license key or license file, press the **Activation** button to enter your license key or browse for the license file.

If you do not yet have a valid license, use the e-mail button to request an evaluation license or to purchase a license.

4. Select **which kind of license** you have and **who** should use the license.



What kind of license did you request?

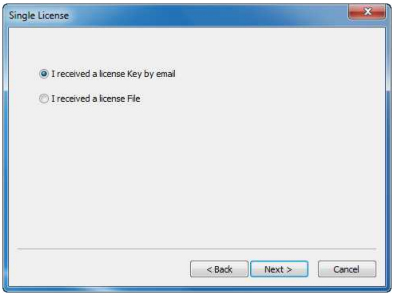
● Single
○ Concurrent (network license)
○ Evaluation

By who should the license be used?

● All Users on this computer
○ Only Me

License installation dialog.

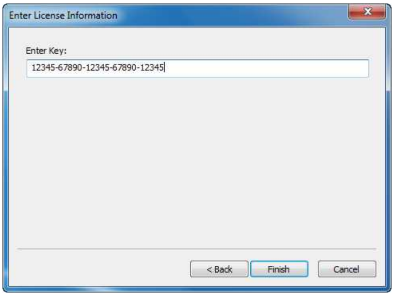
5. On the next dialog, select ***I received a license key by e-mail*** when you have a license key. When you got a **license file**, select the other option.



● I received a license Key by email
○ I received a license File

License key or license file?

6. Enter your **20-sim 4C license key** (or browse for the license file):

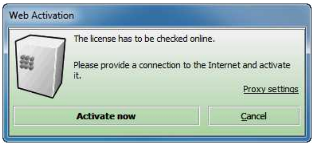


Enter License Information

Enter Key:
12345-67890-12345-67890-12345

Enter your license key.

7. In case of a license key, 20-sim 4C will now validate your license on-line. Please make sure that you have a **working Internet connection**. Click **Activate now** to continue.



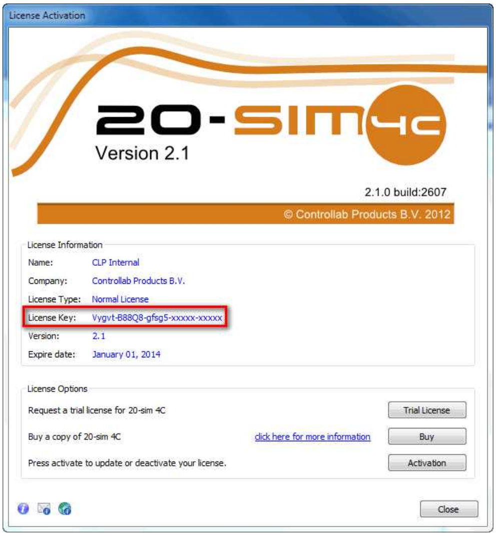
Web Activation

The license has to be checked online.

Please provide a connection to the Internet and activate it.

On-line license activation.

As a result of a successful license installation, 20-sim 4C shows your license information in a dialog:



License Activation

License Information

Name: CLP Internal
Company: Controllab Products B.V.
License Type: Normal License
License Key: Vygvt-B88Q8-gfsg5-xxxxx-xxxxx
Version: 2.1
Expire date: January 01, 2014

License Options

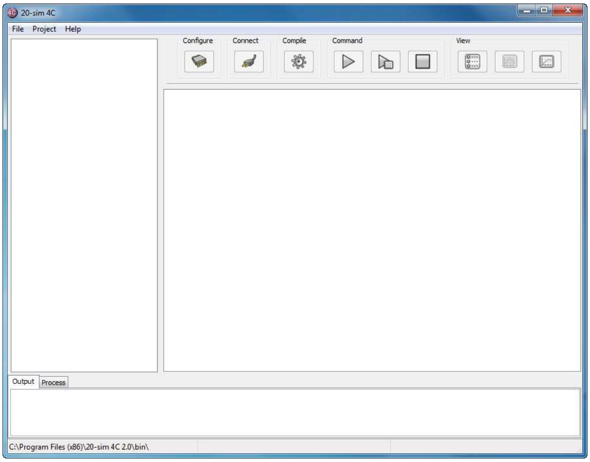
Request a trial license for 20-sim 4C

Buy a copy of 20-sim 4C

Press activate to update or deactivate your license.

License activation was successful.

After closing the license information dialog, 20-sim 4C is ready for use and should look like the figure below:



C:\Program Files (x86)\20-sim 4C 2.0\bin\

20-sim 4C.

If you cannot open 20-sim 4C or have problems with the installation or the license activation, please check the troubleshooting section. If this does not help, please contact Controllab Products B.V.

4.3 Uninstalling 20-sim 4C

You can uninstall 20-sim 4C by clicking the **Uninstall** command from the 20-sim 4C start menu. Uninstallation of 20-sim 4C will not deactivate your license. If you want to move 20-sim 4C to another computer, you have to **deactivate your license** first before uninstalling.

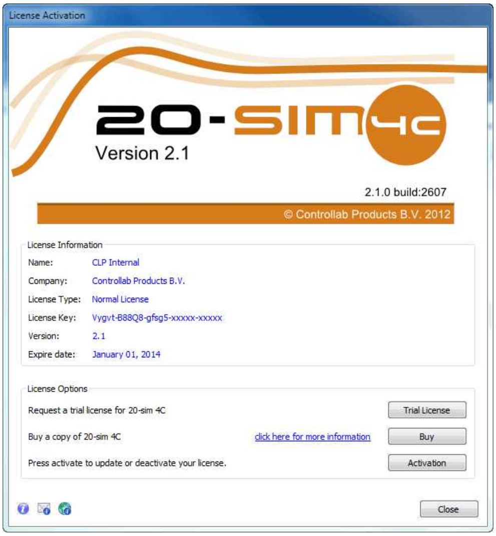
4.4 Deactivation

If you want to move 20-sim 4C to another computer, you have to deactivate your license, before uninstalling the program. On the new computer you can then install the program and activate the license.

To deactivate your license:

1. From the Windows **Start menu** open **20-sim 4C**.
2. From the **Help** menu choose **License Activation**.

The *License Activation* dialog will open.



License Information

Name: CLP Internal

Company: Controllab Products B.V.

License Type: Normal License

License Key: Vygvt-B88Q8-gfsg5-xxxxx-xxxxx

Version: 2.1

Expire date: January 01, 2014

License Options

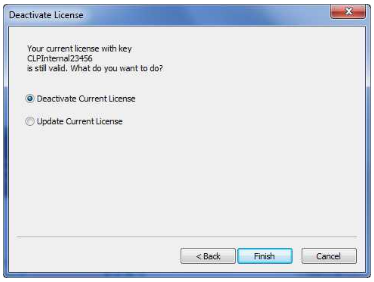
Request a trial license for 20-sim 4C

Buy a copy of 20-sim 4C click here for more information

Press activate to update or deactivate your license.

The License Activation dialog.

3. Click on the **Activation** button and then select **Deactivate Current License**.



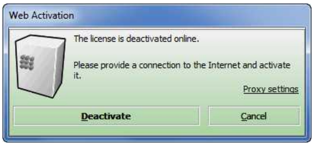
Your current license with key
CLPInternal23456
is still valid. What do you want to do?

● Deactivate Current License

○ Update Current License

Click Deactivate Current License if you want to use the program on another computer.

4. Click **Finish** to deactivate your license. You will be asked to confirm the deactivation and have to click **Deactivate**.



The license is deactivated online.

Please provide a connection to the Internet and activate it.

4.5 Upgrading

If you want to upgrade to a newer version of 20-sim 4C, go to the 20-sim 4C website and download the new version. After downloading you can install it.

- If the new version is a minor update (e.g. version 2.0.x.x), it will automatically replace the old version. After installation you can run the new version immediately. No new license is required.
- If the new version is a major upgrade (e.g. version 2.1.x.x or higher), it will be installed next to the old version. Both versions can be used at the same time. The new version will require a new license.

4.6 Unattended Installation

An unattended installation is an installation that is performed without user interaction during its progress or with no user present at all.
To perform an unattended installation on the default 'program files' installation directory, run the following command :

```
20-sim-4C-2.1.x.x-win32.exe /S
```

It is possible to set an alternative installation directory by specifying the /D argument. It must be the last parameter used in the command line and must not contain any quotes, even if the path contains spaces. Only absolute paths are supported.

```
20-sim-4C-2.1.x.x-win32.exe /S /D=D:\My Installation Files\20-sim 4C 2.1
```

4.7 Connecting the Target

20-sim 4C uses an ethernet connection to communicate with a target. This connection can be established using a cross cable that is connected between your PC (the host PC) and the target directly or by connecting your PC and the target to your local network. Using a cross-cable to connect the host PC directly to the target is recommended if you encounter problems with the connection.

Plug in the cross cable or network cable.

Cross Cable

1. Plug the **cross cable** in the **connector** of your target.

2. The other end of the **cross cable** must be plugged into your **host PC**.

It may take several minutes before Windows has configured the network interface. Please be patient when 20-sim 4C cannot make a connection immediately.

Network Cable

1. Plug the **network cable** in the connector of your target.

2. The other end of the **network cable** must be plugged into a **switch or hub of the network**.

Use a 'normal' network cable to connect the target via a switch or hub to your network. If your network provides a DHCP server the configuration is automatic. If your network does not provide a DHCP server make sure your network is configured in the 192.168.1.x range and that your PC computer has a proper connection to the network.

See Networking in the Troubleshooting chapter in case you experience problems.

4.8 Running a Test model

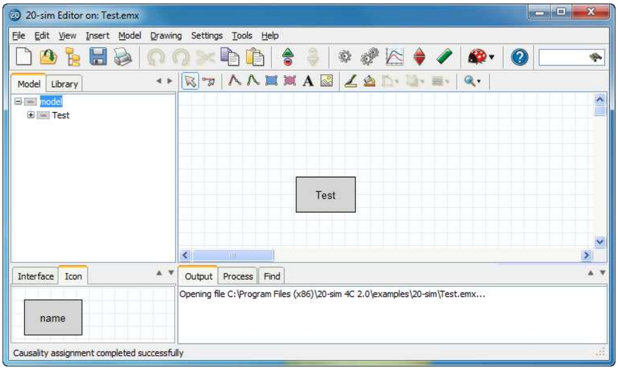
To test if the installation was successful, we will run an example model in 20-sim and generate C-code that will be transferred to 20-sim 4C to run on your target. The model is called *Test.emx* and has no external inputs or outputs.

Exporting the 20-sim Model to 20-sim 4C

1. From the Windows Start menu open **20-sim**.

2. In **20-sim** load the model ***C:\Program Files\20-sim 4C 2.1\examples\20-sim\Test.emx***.

Note: For 64-bit Windows versions the location is: *C:\Program Files (x86)\20-sim 4C 2.1\examples\20-sim\Test.emx*. If your version of 20-sim 4C is installed on a different location, please use the correct path.



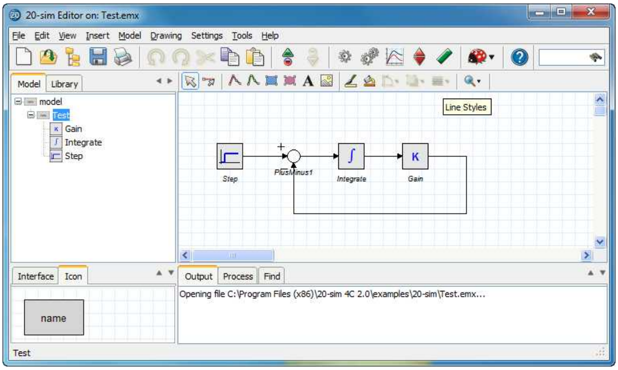
Opening file C:\Program Files (x86)\20-sim 4C 2.0\examples\20-sim\Test.emx...

Causality assignment completed successfully

The model Test.emx loaded in 20-sim.

The 20-sim model Test.emx contains one submodel (named Test).

3. Select the submodel **Test** and from the Model menu select **Go Down**.



Opening file C:\Program Files (x86)\20-sim 4C 2.0\examples\20-sim\Test.emx...

The submodel Test contains a block diagram model with no external inputs or outputs.

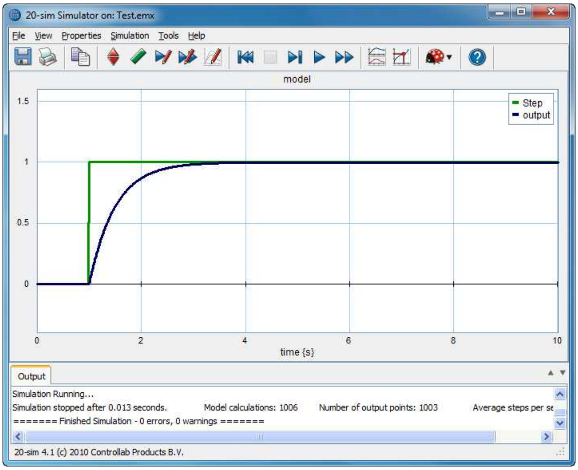
As you can see the submodel Test is a block diagram model of a first order system which contains no external inputs or outputs.

4. From the **Model** menu select **Start Simulator**.

Now the 20-sim Simulator opens showing an empty plot.

5. In the **Simulator** from the **Simulation** menu select **Run** to run a simulation.

You should see a stepwise change in the setpoint signal and a first order response of the system.


20-sim Simulator on: Test.emx
File View Properties Simulation Tools Help
model
1.5
1
0.5
0
0
2
4
6
8
10
time {s}
Step
output
Output
20-sim 4.1 (c) 2010 Controllab Products B.V.

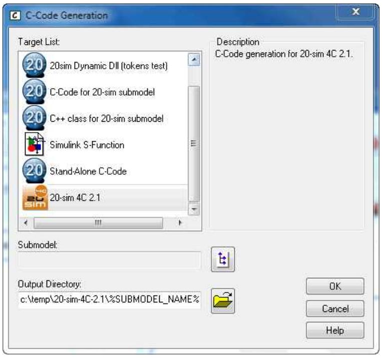
```
Simulation Running...
Simulation stopped after 0.013 seconds.        Model calculations: 1006       Number of output points: 1003        Average steps per se
======= Finished Simulation - 0 errors, 0 warnings =======
```

The simulation results for the model Test.emx.

We will generate C-code from this model and run it on a target.

6. In the Simulator, from the **Tools** menu select **Real Time Toolbox** and **C-code Generation**. The C-code Generation dialog will pop-up.

7. Select the **20-sim 4C target**. Make sure it looks like the figure below (in 20-sim 4.1 multiple targets may be shown). If the 20-sim 4C target is not shown, please check the troubleshooting section.



C-Code Generation

Target List:

- 20sim Dynamic Dll (tokens test)
- C-Code for 20-sim submodel
- C++ class for 20-sim submodel
- Simulink S-Function
- Stand-Alone C-Code
- 20-sim 4C 2.1

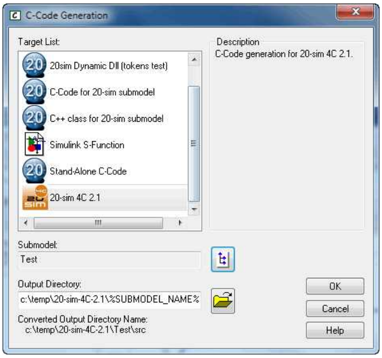
Description: C-Code generation for 20-sim 4C 2.1.

Submodel:

Output Directory: c:\temp\20-sim-4C-2.1\%SUBMODEL_NAME%

The C-code generation menu in 20-sim.

8. Click on the **Submodel button** and select the submodel *Test*. The C-code Generation dialog should look like:



C-Code Generation

Target List:

- 20sim Dynamic Dll (tokens test)
- C-Code for 20-sim submodel
- C++ class for 20-sim submodel
- Simulink S-Function
- Stand-Alone C-Code
- 20-sim 4C 2.1

Description: C-Code generation for 20-sim 4C 2.1.

Submodel: Test

Output Directory: c:\temp\20-sim-4C-2.1\%SUBMODEL_NAME%

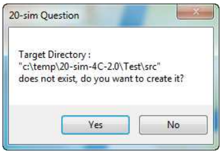
Converted Output Directory Name: c:\temp\20-sim-4C-2.1\Test\src

The C-code generation menu in 20-sim.

Note that a default location for the C-code files is given in the Output Directory field: *C:\temp\20sim-4C\%SUBMODEL_NAME%*, where *%SUBMODEL_NAME%* will be replaced by the submodel name. If you want to use a different location, please enter it here.

9. Click the **OK** button to close the dialog.

Now the model will be exported to 20-sim 4C as C-code. All code files are stored in a temporary folder. A dialog may appear asking you to create this folder.

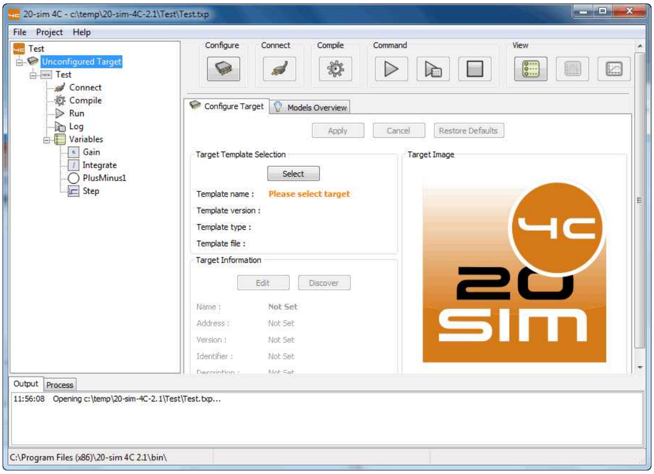


Target Directory :
"c:\temp\20-sim-4C-2.0\Test\src"
does not exist, do you want to create it?

The folder for temporary code files.

10. Click **Yes** to create the folder (or go to step 7 to enter another directory).

If everything works fine, 20-sim 4C will be opened, with the Test model loaded:



Template name : Please select target
Template version :
Template type :
Template file :

Name : Not Set
Address : Not Set
Version : Not Set
Identifier : Not Set
Description : Not Set

```
11:56:08   Opening c:\temp\20-sim-4C-2.1\Test\Test.txp...
```

C:\Program Files (x86)\20-sim 4C 2.1\bin\

20-sim 4C with the Test model loaded.

If an error occurs and 20-sim 4C does not open, please check the troubleshooting section.

20-sim 4C Settings and Monitoring

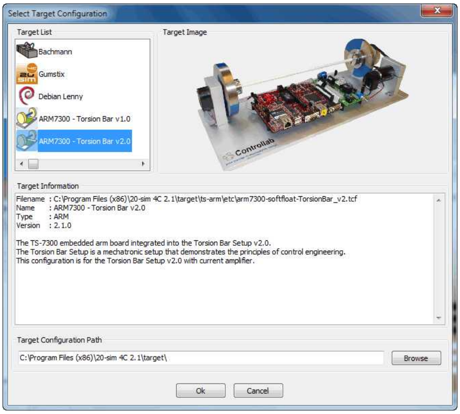
We will now enter the settings to get the model running on your target and choose the variables that we would like to monitor.

Configure

1. Click on the **Configure** button .

2. Click on the **Select** button, select the proper target and click OK.

The next picture shows as an example the ARM7300 - Torsion Bar v2.0 target.



Select Target Configuration
Target List
Bachmann
Gumstix
Debian Lenny
ARM7300 - Torsion Bar v1.0
ARM7300 - Torsion Bar v2.0
Target Image
Controllab
Target Information
Target Configuration Path
Browse
Ok
Cancel

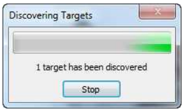
```
Filename  : C:\Program Files (x86)\20-sim 4C 2.1\target\ts-arm\etc\arm7300-softfloat-TorsionBar_v2.tcf
Name      : ARM7300 - Torsion Bar v2.0
Type      : ARM
Version   : 2.1.0

The TS-7300 embedded arm board integrated into the Torsion Bar Setup v2.0.
The Torsion Bar Setup is a mechatronic setup that demonstrates the principles of control engineering.
This configuration is for the Torsion Bar Setup v2.0 with current amplifier.
```

C:\Program Files (x86)\20-sim 4C 2.1\target\

Select the proper target.

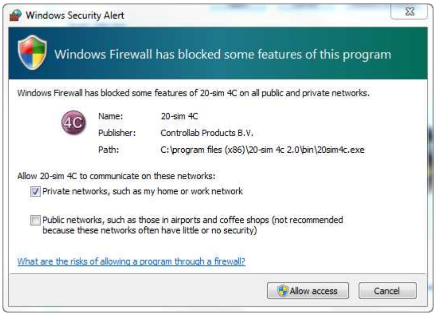
20-sim 4C will now start a search to find the IP-addresses of connected targets:



1 target has been discovered

Finding the IP-address.

If a firewall is installed, a warning message may pop-up:



Windows Firewall has blocked some features of this program

Windows Firewall has blocked some features of 20-sim 4C on all public and private networks.

Name: 20-sim 4C
Publisher: Controllab Products B.V.
Path: C:\program files (x86)\20-sim 4c 2.0\bin\20sim4c.exe

Allow 20-sim 4C to communicate on these networks:

☑ Private networks, such as my home or work network

□ Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

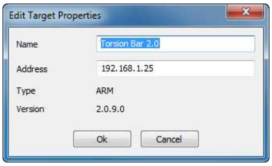
What are the risks of allowing a program through a firewall?

The Windows firewall message when the Refresh button is clicked.

3. Click the **Allow Access** button (Windows 7) or the **Unblock** (Windows XP, Windows Vista) button (or a similar button if another firewall message is shown) if the name of the program is 20-sim 4C and the publisher is Controllab Products B.V.

4. *Optional:* Press the **Discover** button on the **Configure Target** tab to manually repeat step number 3.

5. *Optional:* You can change the target name and IP-address manually for your target using the **Edit** button.

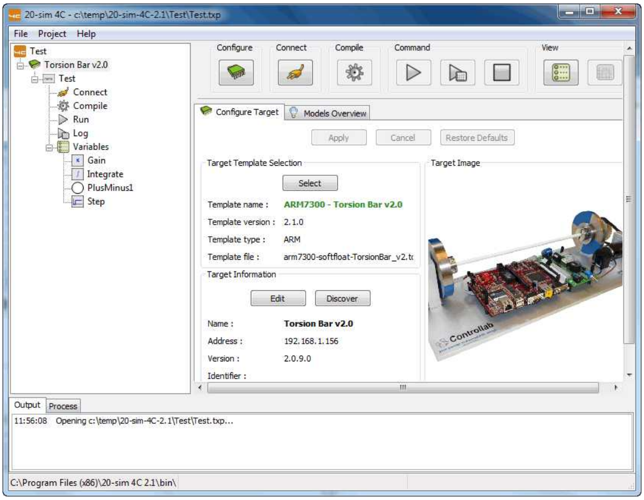


Name: Torsion Bar 2.0
Address: 192.168.1.25
Type: ARM
Version: 2.0.9.0

Changing the target name and IP-address manually.

6. Click on the **Apply** button.

If everything was installed and connected successfully, the Configure button should turn into green .



Template name : ARM7300 - Torsion Bar v2.0
Template version : 2.1.0
Template type : ARM
Template file : arm7300-softfloat-TorsionBar_v2.t

Name : Torsion Bar v2.0
Address : 192.168.1.156
Version : 2.0.9.0
Identifier :

```
11:56:08   Opening c:\temp\20-sim-4C-2.1\Test\Test.txp...
```

C:\Program Files (x86)\20-sim 4C 2.1\bin\

The configuration was entered successfully.

If the Configure button has turned to red please check the Troubleshooting section.

Connect

7. Click on the **Connect** button .

8. Our model does not have external inputs or outputs that we have to connect so click the **Apply** button.

Compile

9. Click on the Compile button to compile the generated code into an executable.

The Compile button should turn into green when the compilation process is successfully finished. If it has turned into red please check the Troubleshooting section for help.

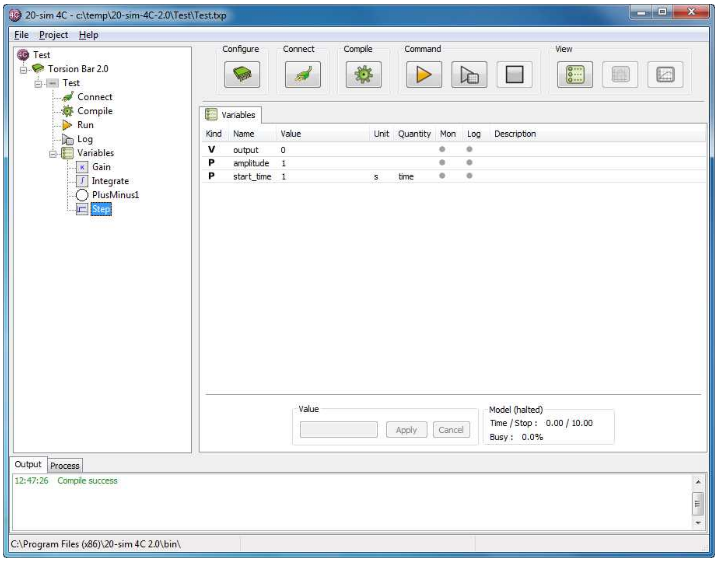
Command

Our model has no external inputs or outputs. To make sure its runs properly we have to monitor its internal variables during running. That is why we will first choose the variables that we will monitor and then run the model.

10. In the tree-view at the left side of the 20-sim 4C windows click **Variables**. It will expand showing the available submodels

11. Click on the ***Step*** submodel.

Now a list of two parameters (amplitude and start_time) and one variable (output) should be visible.



Kind	Name	Value	Unit	Quantity	Mon	Log	Description
V	output	0			●	●	
P	amplitude	1			●	●	
P	start_time	1	s	time	●	●	

Value

Model (halted)
Time / Stop : 0.00 / 10.00
Busy : 0.0%

```
12:47:26 Compile success
```

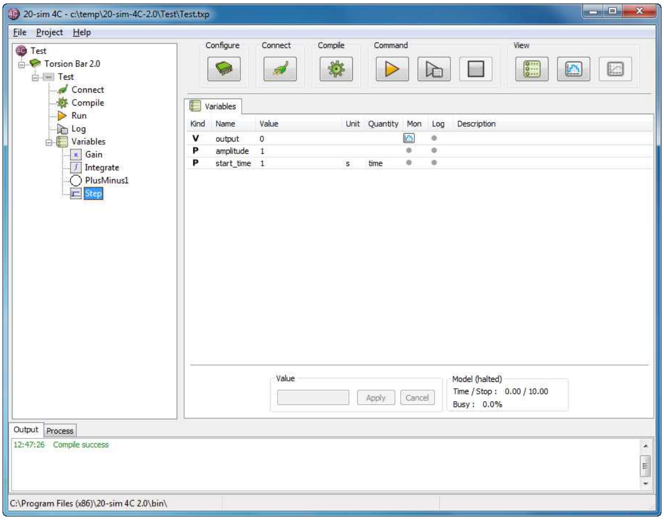
C:\Program Files (x86)\20-sim 4C 2.0\bin\

The list of parameters and variables of the submodel Step.

The variable *output* has two gray dots (Mon and Log) at the left. You can click these dots to select the variable for monitoring or logging.

12. With your mouse pointer go to the variable ***output*** and click on the gray dot to select it for **monitoring**.

20-sim 4C should now look like:



Kind	Name	Value	Unit	Quantity	Mon	Log	Description
V	output	0			📈	●	
P	amplitude	1			●	●	
P	start_time	1	s	time	●	●	

Value

Apply Cancel

Model (halted)
Time / Stop : 0.00 / 10.00
Busy : 0.0%

```
12:47:26   Compile success
```

C:\Program Files (x86)\20-sim 4C 2.0\bin\

The list of parameters and variables of the submodel Step.

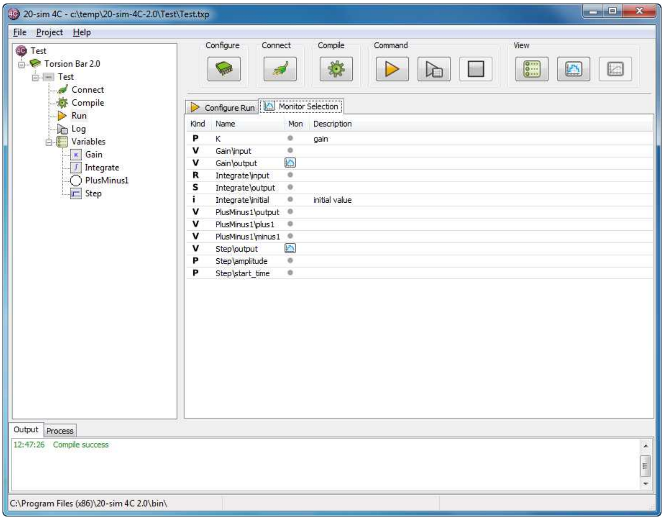
Now we will choose a second variable for monitoring.

13. Click on the ***Gain*** submodel and select the variable ***output*** for **monitoring**.

To see all variables that have been chosen for monitoring, you can view a Monitor Selection.

14. In the tree-view click **Run** and then the tab **Monitor Selection**.

Now 20-sim 4C should look like:



Kind	Name	Mon	Description
P	K	●	gain
V	Gain\input	●	
V	Gain\output	📈	
R	Integrate\input	●	
S	Integrate\output	●	
i	Integrate\initial	●	initial value
V	PlusMinus1\output	●	
V	PlusMinus1\plus1	●	
V	PlusMinus1\minus1	●	
V	Step\output	📈	
P	Step\amplitude	●	
P	Step\start_time	●	

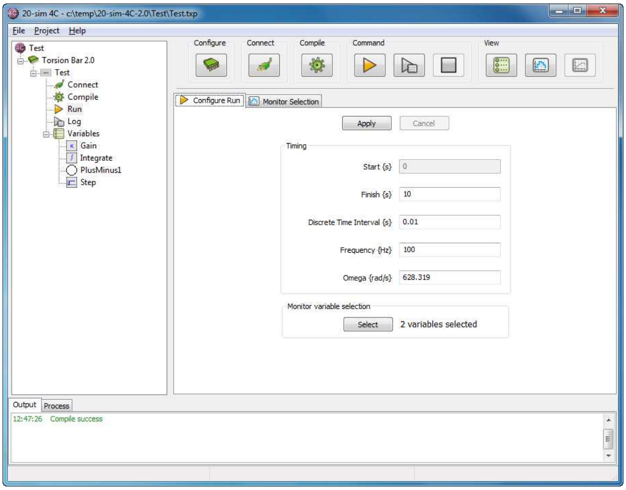
```
12:47:26   Compile success
```

C:\Program Files (x86)\20-sim 4C 2.0\bin\

The list of monitored variables.

Now we will configure the run settings, open a monitor plot and start to run the model on the target:

15. Select the **Configure Run** tab and click on the **Apply** button to accept the default settings for this run experiment. The **Run** button should become green .



Timing

Start {s} 0

Finish {s} 10

Discrete Time Interval {s} 0.01

Frequency {Hz} 100

Omega {rad/s} 628.319

Monitor variable selection

Select 2 variables selected

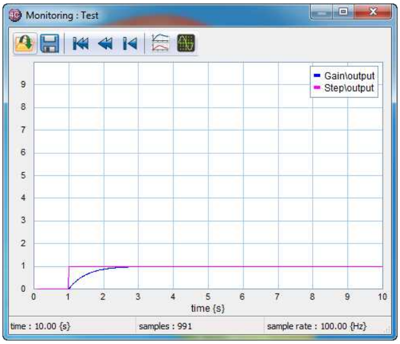
```
12:47:26   Compile success
```

By default, 20-sim 4C will use the same settings as used in the 20-sim simulator. On the Configure Run tab, you can change for example the Finish time for your model.

16. Click on the **Monitor** button to open a plot window.

17. Click on the **Run** button to start running the model on the target.

Now you should see the following plot after 10 seconds of running. The plot shows a stepwise change in the setpoint signal and a first order response of the system.



time : 10.00 {s}   samples : 991   sample rate : 100.00 {Hz}

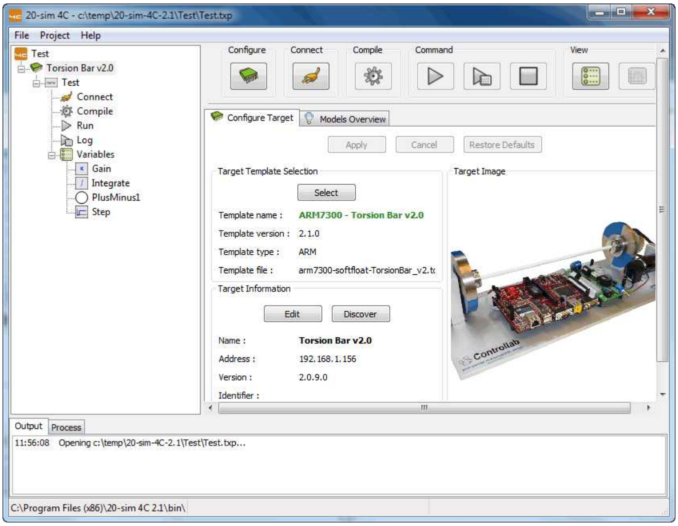
The monitored variables, shown in a plot.

If your results are similar, 20-sim 4C is properly installed and your target is properly working. If the plot results are not similar but some other variable is shown in the plot, check if the correct variables were chosen for monitoring (hint: check the Monitor Selection tab of step 14).

5 Working with 20-sim 4C

5.1 Introduction

20-sim 4C is a prototyping environment that allows you to export C-code to hardware like real-time PC systems and embedded real-time Linux processor boards. The name 4C stands for *Configure*, *Connect*, *Compile* and *Command* and these are the tasks that you have to perform to get code running on a target. The main window of 20-sim 4C shows these tasks at the top of the window.



Template name : ARM7300 - Torsion Bar v2.0
Template version : 2.1.0
Template type : ARM
Template file : arm7300-softfloat-TorsionBar_v2.t(

Name : Torsion Bar v2.0
Address : 192.168.1.156
Version : 2.0.9.0
Identifier :

```
11:56:08   Opening c:\temp\20-sim-4C-2.1\Test\Test.txp...
```

C:\Program Files (x86)\20-sim 4C 2.1\bin\

20-sim 4C helps you to get 20-sim models running as C-code on real-time processors.

Model

The collection of C-code files that will be exported to the target is called a model, like the model in 20-sim. The model is shown in a tree at the left of the main window as part of the selected target. The tree also shows configuration properties for the model: I/O connections, run settings, data logging settings, available variables for logging and monitoring and configurable model parameters.

Configure

In the configuration step, you can select the target platform for your generated code. After the target selection, 20-sim 4C will automatically search on the network for your target and connect to it to retrieve its current status.

Connect

Most targets have connections to onboard and external devices (e.g. sensors, actuators or other I/O devices). In the connect step, you have to connect the inputs and outputs of your model to the proper target devices.

Compile

To get the C-code running on a target, a compiler is used to create executable code.

Command

With the command buttons you can run the executable code on a target, stop it, monitor and log variables.

View

Use the view buttons to show variables and plots.

5.2 Working Order

20-sim

You can import your own C-code in 20-sim 4C, but the easiest way to get code running on a target is to generate it with 20-sim. 20-sim 4C will be opened automatically with the C-code loaded.

1. Open **20-sim** and load your model.
2. Start the **Simulator**.
3. In the Simulator from the **Tools** menu select the **Real Time Toolbox** command and **C-code Generation**.
4. Select the **20-sim 4C** Target and select the submodel to generate code from.
5. Click **OK** to generate the C-code.

20-sim 4C will automatically start and the generated C-code will be loaded.

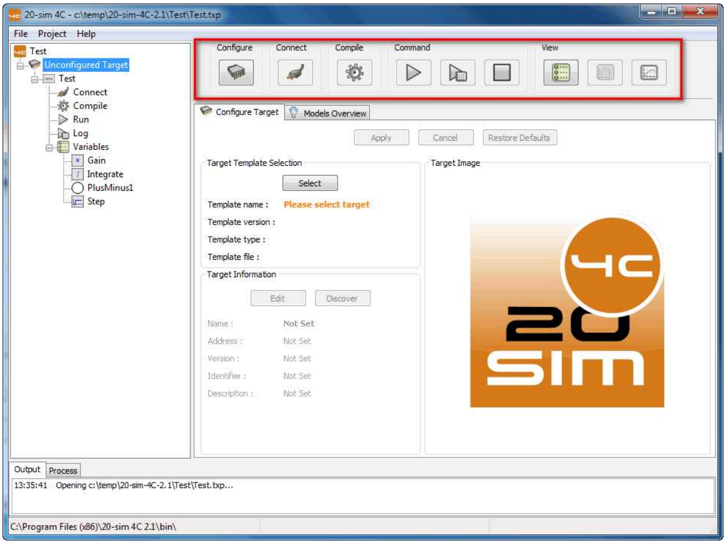
20-sim 4C

The name 4C stands for Configure, Connect, Compile and Command and that is the correct working order to get code running on external targets. You can click the buttons at the top of the window in the *button bar* from left to right.

1. Configure: specify target specific settings.
2. Connect: connect model inputs and outputs with hardware inputs and outputs.
3. Compile: compile the model (C-code) for the target.
4. Command: run the compiled model and do e.g. measurements.

All the buttons and icons use color to reflect the state of the tasks. The following colors are used:

- Grey: The item is currently not available.
- Red: The item has failed an action. (See the output or process log at the bottom of the screen for details.)
- Orange: The item is out of date and needs user action.
- Green: The item is ok and needs no further attention.



Template name : Please select target
Template version :
Template type :
Template file :

Name : Not Set
Address : Not Set
Version : Not Set
Identifier : Not Set
Description : Not Set

```
13:35:41   Opening c:\temp\20-sim-4C-2.1\Test\Test.txp...
```

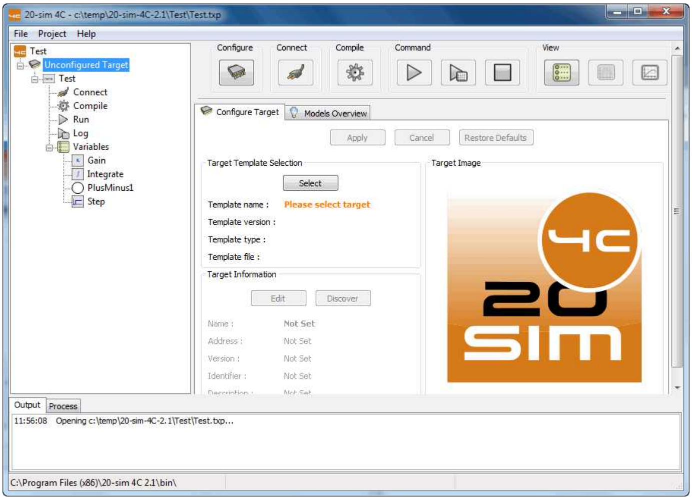
C:\Program Files (x86)\20-sim 4C 2.1\bin\

Click the buttons at the top of the window, from left to right.

5.3 Configure

In the Configure step, you can select a new (or change to a different) target for your model to run on. The Configure step has two tabs, the Configure Target tab and a Models Overview tab.

Configure Target



Template name : Please select target
Template version :
Template type :
Template file :

Name : Not Set
Address : Not Set
Version : Not Set
Identifier : Not Set
Description : Not Set

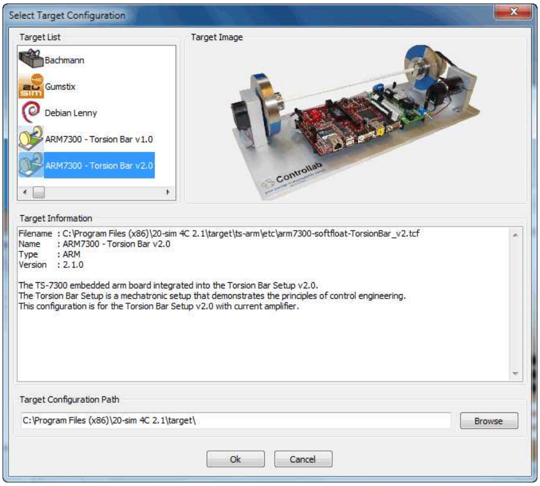
```
11:56:08   Opening c:\temp\20-sim-4C-2.1\Test\Test.txp...
```

C:\Program Files (x86)\20-sim 4C 2.1\bin\

Select the target and enter the IP address.

1. First select the desired target by pressing the **Select** button.

A dialog will be opened showing a list of supported targets. If desired, you create your own targets. The *Filename* shows the location of the corresponding Target Specification files.



Select Target Configuration
Target List
Bachmann
Gumstix
Debian Lenny
ARM7300 - Torsion Bar v1.0
ARM7300 - Torsion Bar v2.0
Target Image
Controllab
Target Information

Filename : C:\Program Files (x86)\20-sim 4C 2.1\target\ts-arm\etc\arm7300-softfloat-TorsionBar_v2.tcf
Name : ARM7300 - Torsion Bar v2.0
Type : ARM
Version : 2.1.0

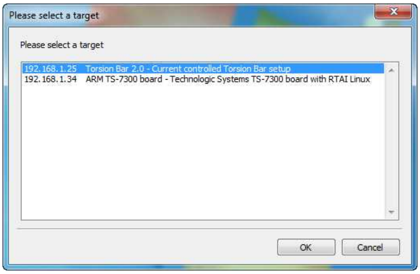
The TS-7300 embedded arm board integrated into the Torsion Bar Setup v2.0.
The Torsion Bar Setup is a mechatronic setup that demonstrates the principles of control engineering.
This configuration is for the Torsion Bar Setup v2.0 with current amplifier.

Target Configuration Path: C:\Program Files (x86)\20-sim 4C 2.1\target\

Select the proper target.

2. **Select** your target and close the dialog by pressing **OK**.

20-sim will automatically discover connected targets and show a list of found targets with their names and IP-addresses. A target selection dialog will be shown if 20-sim 4C detects more than one 4C compatible target:

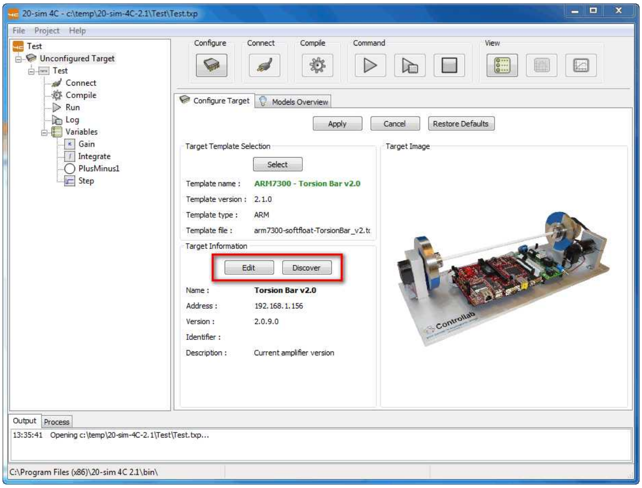


192.168.1.25 Torsion Bar 2.0 - Current controlled Torsion Bar setup
192.168.1.34 ARM TS-7300 board - Technologic Systems TS-7300 board with RTAI Linux

Target selection dialog when multiple targets are detected.

If the automatic discovery of target fails, you can:

3. Click the **Discover** button to do another attempt to discover targets.
4. Click the **Edit** button to enter the target name and IP-address manually.



Template name : ARM7300 - Torsion Bar v2.0
Template version : 2.1.0
Template type : ARM
Template file : arm7300-softfloat-TorsionBar_v2.tc

Name : Torsion Bar v2.0
Address : 192.168.1.156
Version : 2.0.9.0
Identifier :
Description : Current amplifier version

```
13:35:41   Opening c:\temp\20-sim-4C-2.1\Test\Test.txp...
```

C:\Program Files (x86)\20-sim 4C 2.1\bin\

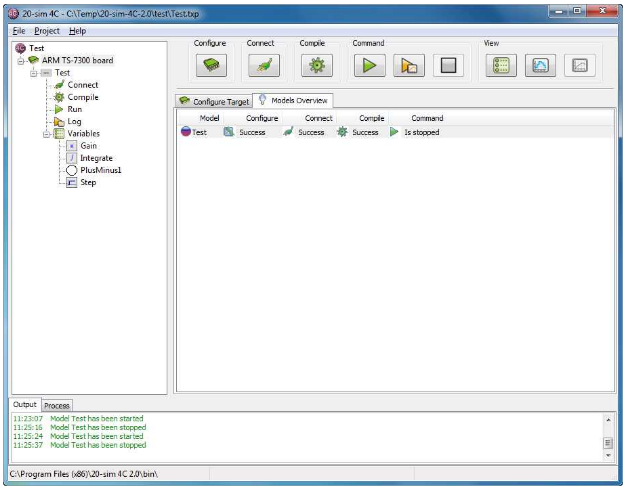
Use the Discover or Edit buttons if the automatic connection fails.

5. Click the **Apply** button on the **Configure Target** tab to apply the target settings.

If the connection cannot be made, the *Configure* button will be shown with a red color . If the connection was successful, the *Configure* button will be shown with a green color . Normally every second the target is polled to retrieve its state.

Models overview

The *Models Overview* tab shows the status of the models that are assigned to a particular target:



Model	Configure	Connect	Compile	Command
Test	Success	Success	Success	Is stopped

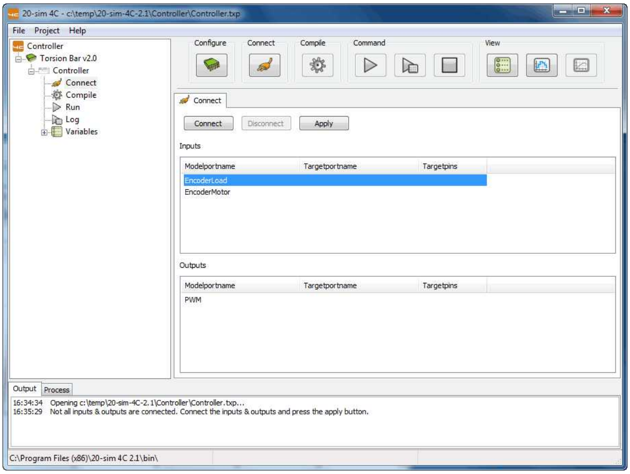
```
11:23:07   Model Test has been started
11:25:16   Model Test has been stopped
11:25:24   Model Test has been started
11:25:37   Model Test has been stopped
```

C:\Program Files (x86)\20-sim 4C 2.0\bin\

The Models Overview.

5.4 Connect

Most targets can read sensor data, steer actuators or communicate via other I/O facilities and buses (e.g. a CAN bus) with the environment. You have to connect the inputs and outputs of the model with the inputs and outputs of the selected target. An example is shown in the figure below. The model has two inputs (*EncoderLoad* and *EncoderMotor*) and one output (*PWM*). We have to choose the matching input and output of the target.



Inputs

Modelportname	Targetportname	Targetpins
EncoderLoad		
EncoderMotor		

Outputs

Modelportname	Targetportname	Targetpins
PWM		

```
16:34:34   Opening c:\temp\20-sim-4C-2.1\Controller\Controller.txp...
16:35:29   Not all inputs & outputs are connected. Connect the inputs & outputs and press the apply button.
```

C:\Program Files (x86)\20-sim 4C 2.1\bin\

Connect the inputs and outputs of a model with the inputs and outputs of the target.

Inputs

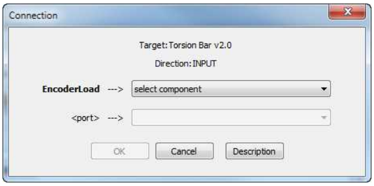
In the inputs list under *Modelportname* you will see a list of inputs.

1. **Select** one of the **inputs** (mouse pointer on top and click the left mouse button).

Now the Connect button will be active.

2. Click the **Connect** button.

The *Connect* dialog will appear. In this dialog you can select one out of a set of matching hardware inputs.



Target: Torsion Bar v2.0

Direction: INPUT

EncoderLoad ---> select component

<port> --->

Use the Connect dialog to choose a matching hardware input or output.

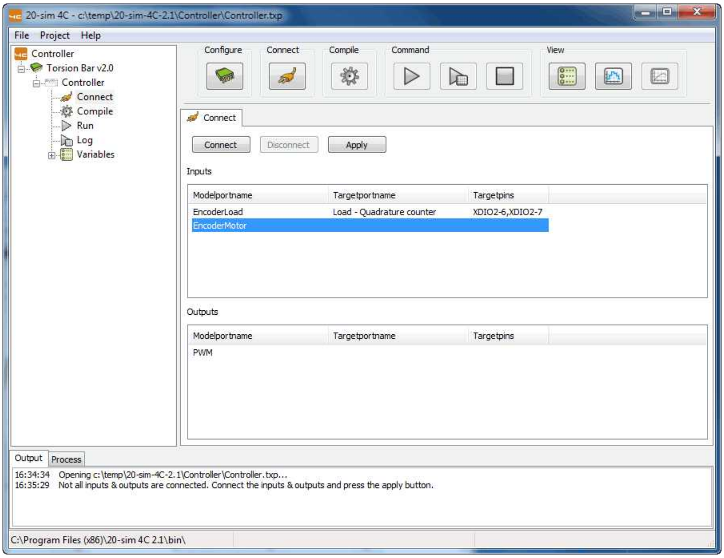
3. Use the **select component** drop down box to select the type of hardware input to connect with.

If there is only one hardware input, the second drop down box will be empty. If there are multiple hardware inputs, you must specify in the <port> drop down box which port you want to connect. An example of an input component is an Analog to Digital (AD) converter. This converter could have 4 individual inputs (ports). To connect a model input to a particular AD channel, you need to select first the AD converter component and then the right port.

4. Use the **<port>** drop down box to connect the input to a specific port.

5. Click **OK** to close the *Connect* dialog.

In the inputs list you will see the model input and the connected hardware input. Under Modelportname will find the model inputs. Under Targetportname the connected hardware inputs are shown. Under Targetpins the physical address of the hardware outputs are shown (you can use this name to find the correct pin at the hardware layout).



Inputs

Modelportname	Targetportname	Targetpins
EncoderLoad	Load - Quadrature counter	XDIO2-6,XDIO2-7
EncoderMotor		

Outputs

Modelportname	Targetportname	Targetpins
PWM		

```
16:34:34   Opening c:\temp\20-sim-4C-2.1\Controller\Controller.txp...
16:35:29   Not all inputs & outputs are connected. Connect the inputs & outputs and press the apply button.
```

The model input is now connected with a hardware input.

6. If you have **multiple model inputs**, you have to repeat these actions until all inputs are connected.

Outputs

7. The **model outputs** have to be matched with **hardware outputs** in a similar way as the inputs.

8. When all inputs and outputs are connected, click the **Apply** button.

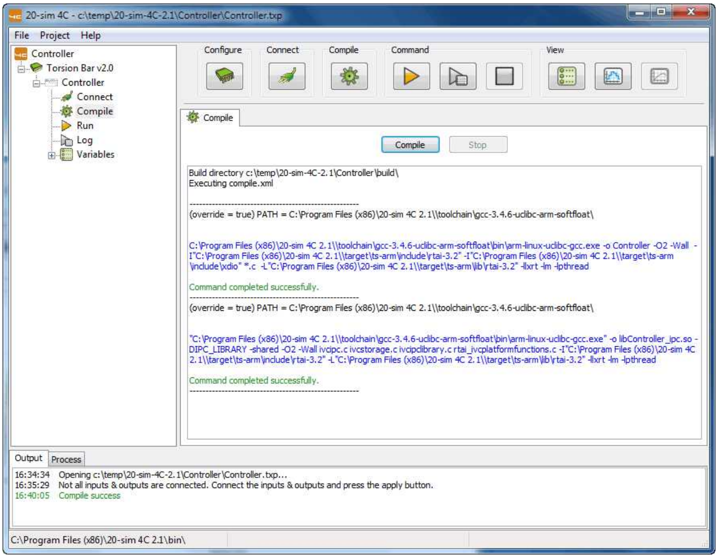
20-sim 4C will now extend the generated C-code with the target-specific C-code to couple the model inputs and output to the just selected hardware ports. The *Connect* button will change from orange into green . If an error occurs the *Connect* button will turn into red. The Process tab at the bottom of the window can be selected to view more detailed information.

5.5 Compile

If the Configuration and Connection were successful, the *Configure* button and the *Connect* button are shown in green. Now we can start the cross compiler to convert the model (the C-code files) into target executable code.

1. Click the **Compile** button.

If the compilation is successful, the Compile button will change from orange into green. If an error occurred, the *Compile* button will change into red. You should then check the compilation log for compiler errors. See the Troubleshooting section for more detailed help on compilation errors.



```
Build directory c:\temp\20-sim-4C-2.1\Controller\build\
Executing compile.xml

------------------------------------------------------
(override = true) PATH = C:\Program Files (x86)\20-sim 4C 2.1\\toolchain\gcc-3.4.6-uclibc-arm-softfloat\


C:\Program Files (x86)\20-sim 4C 2.1\\toolchain\gcc-3.4.6-uclibc-arm-softfloat\bin\arm-linux-uclibc-gcc.exe -o Controller -O2 -Wall -
I"C:\Program Files (x86)\20-sim 4C 2.1\\target\ts-arm\include\rtai-3.2" -I"C:\Program Files (x86)\20-sim 4C 2.1\\target\ts-arm
\include\xdio" *.c  -L"C:\Program Files (x86)\20-sim 4C 2.1\\target\ts-arm\lib\rtai-3.2" -llxrt -lm -lpthread

Command completed successfully.
------------------------------------------------------
(override = true) PATH = C:\Program Files (x86)\20-sim 4C 2.1\\toolchain\gcc-3.4.6-uclibc-arm-softfloat\


"C:\Program Files (x86)\20-sim 4C 2.1\\toolchain\gcc-3.4.6-uclibc-arm-softfloat\bin\arm-linux-uclibc-gcc.exe" -o libController_ipc.so -
DIPC_LIBRARY -shared -O2 -Wall ivcipc.c ivcstorage.c ivcipclibrary.c rtai_ivcplatformfunctions.c -I"C:\Program Files (x86)\20-sim 4C
2.1\\target\ts-arm\include\rtai-3.2" -L"C:\Program Files (x86)\20-sim 4C 2.1\\target\ts-arm\lib\rtai-3.2" -llxrt -lm -lpthread

Command completed successfully.
------------------------------------------------------
```

```
16:34:34   Opening c:\temp\20-sim-4C-2.1\Controller\Controller.txp...
16:35:29   Not all inputs & outputs are connected. Connect the inputs & outputs and press the apply button.
16:40:05   Compile success
```

```
C:\Program Files (x86)\20-sim 4C 2.1\bin\
```

Click the Compile button to compile the model into target executable code.

5.6 Command

Once the code has been compiled successfully, you are ready to transfer it to the selected target and run it. The command step has three command buttons (at the top of the window):

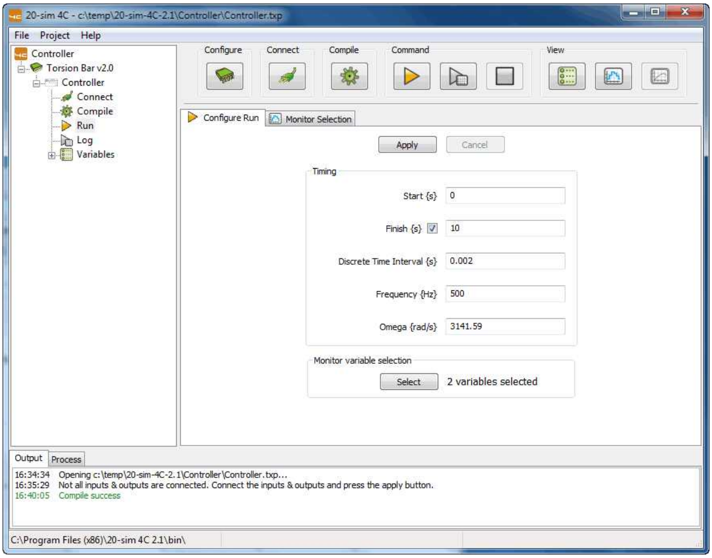
- **Run**
- **Run with logging**
- **Stop**

After the compilation step, the Run button is still orange which means that 20-sim 4C needs additional configuration information for the run before it can run the model on the selected target. Click on the orange Run button to select the *Configure Run* tab to configure the run-time settings.

Configure Run

The model can run for a specific time at a specific rate (the sample frequency). By default, 20-sim 4C will use the same settings as used in the 20-sim Simulator. On the Configure Run tab, you can change for example:

- The **Finish time** of your model: You can either choose to set a finish time by checking the check box or uncheck the check box for long time run.
- The **sample frequency** (change the *Discrete Time Interval*, the *Frequency* or *Omega*).



Timing

Start {s}: 0

Finish {s} ☑: 10

Discrete Time Interval {s}: 0.002

Frequency {Hz}: 500

Omega {rad/s}: 3141.59

Monitor variable selection

Select: 2 variables selected

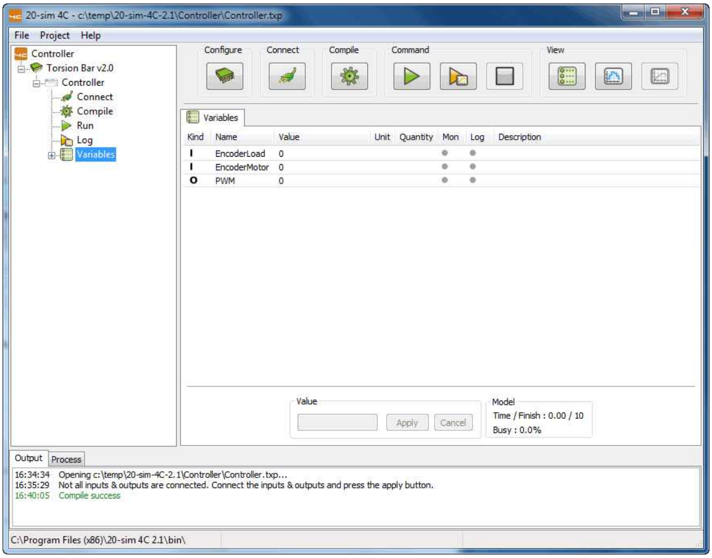
```
16:34:34   Opening c:\temp\20-sim-4C-2.1\Controller\Controller.txp...
16:35:29   Not all inputs & outputs are connected. Connect the inputs & outputs and press the apply button.
16:40:05   Compile success
```

Specify how long the model should run at which rate.

Click the *Apply* button after the settings have been checked and/or altered. The *Run* button will change from orange into green ▷.

Run

The first of the Command buttons is *Run* button ▷. If you click the *Run* button the model is transferred to the target and starts running. The *Run* button changes to grey ▷ and is disabled. The *Stop* button ■ will now become active and green and you can click it to stop the running model. Furthermore, the Variables view will be opened where you can inspect the actual values of inputs, outputs, parameters and variables of the running model.



Kind	Name	Value	Unit	Quantity	Mon	Log	Description
I	EncoderLoad	0			●	●	
I	EncoderMotor	0			●	●	
O	PWM	0			●	●	

Model
Time / Finish : 0.00 / 10
Busy : 0.0%

```
16:34:34   Opening c:\temp\20-sim-4C-2.1\Controller\Controller.txp...
16:35:29   Not all inputs & outputs are connected. Connect the inputs & outputs and press the apply button.
16:40:05   Compile success
```

C:\Program Files (x86)\20-sim 4C 2.1\bin\

Use the Run, Log and Stop buttons to control the model running on a target.

Run with Logging

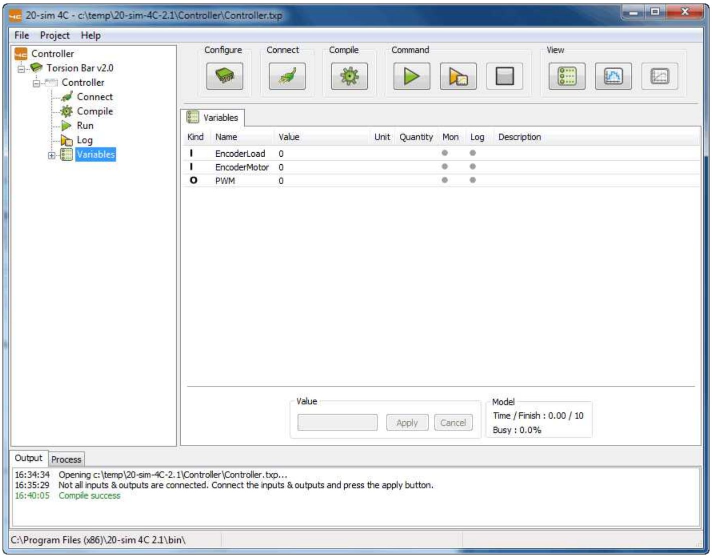
The second button of the Command buttons is *Run with Logging* button . If the button is orange the logger needs to be configured. Pressing the button will automatically open the logger configuration tab. If the button is green the model is transferred to the target and logging is started. After the logging has completed a data file is written to disk and depending on the configuration a plot will be shown. For more information on Logging see Monitoring and Logging.

Stop

You can use the Stop button to stop a running model.

5.7 Variables

When a model is running on the target the Variables view is active. At the bottom of the window you can see the elapsed time, the stop time and the busy percentage of the model. The busy percentage indicates how long the model calculations and I/O access take, compared to the selected sample time. The Variables view shows all variables of a model as well as Change Parameters on-line. Furthermore in the *Variables view*, variables can be selected for monitoring and logging.



Kind	Name	Value	Unit	Quantity	Mon	Log	Description
I	EncoderLoad	0			●	●	
I	EncoderMotor	0			●	●	
O	PWM	0			●	●	

Model
Time / Finish : 0.00 / 10
Busy : 0.0%

```
16:34:34   Opening c:\temp\20-sim-4C-2.1\Controller\Controller.txp...
16:35:29   Not all inputs & outputs are connected. Connect the inputs & outputs and press the apply button.
16:40:05   Compile success
```

C:\Program Files (x86)\20-sim 4C 2.1\bin\

Use the Run, Log and Stop buttons to control the model running on a target.

List of variables

The *Variables* list contains the following columns:

- **Kind:** The kind of the variable, see below.
- **Name**: This column displays the hierarchical name (reflects model structure) of the signal.
- **Value**: The column displays the value of the signal. The value is updated every second when the model is running on the target. If the color of the value is blue the value has been changed but not been applied to the target yet. See Change Parameters for more details.
- **Unit**: This column displays the unit of the variable (when available).
- **Quantity:** This column displays the quantity of the variable (when available).
- **Mon:** You can click on the grey dot ● to select the signal for monitoring. The dot will change into the monitor icon . When a model is running, the variables that have been chosen for monitoring, are shown. You can click the Monitor button at the bottom of the window, to show a plot of the monitored variables. See Run with Monitoring for more details.
- **Log**: You can click on the dot ● to select the signal for logging. The dot will change into the logging icon . See for Run with Logging more details.
- **Description**: This column displays the the description of the variable as given in 20-sim (when available).

You can sort the variables in the Variables list by clicking on the column title of one of the columns.

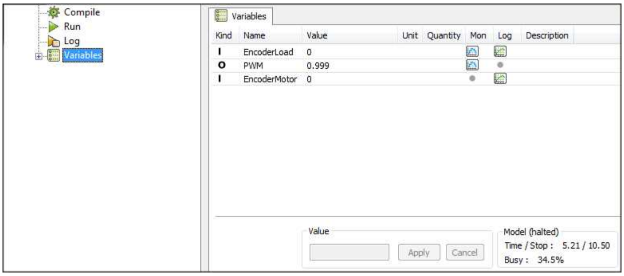
Kind

The following kinds are available in the Variables list:

- **I** = Input
- **O** = Output
- **C** = Constant
- **V** = Variable
- **P** = Parameter
- **A** = Alias (reference to another signal with a different name)

Hierarchy

If a model contains hierarchy (i.e. the model contains submodels which themselves may contain submodels etc.), the list will only show the variables and parameters of the top of the model and global variables and parameters. In the tree-view, the Variables node will then show a plus sign.

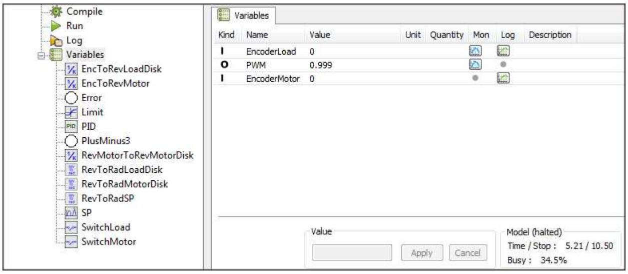


Kind	Name	Value	Unit	Quantity	Mon	Log	Description
I	EncoderLoad	0			📈	📈	
O	PWM	0.999			📈	●	
I	EncoderMotor	0			●	📈	

Model (halted)
Time / Stop : 5.21 / 10.50
Busy : 34.5%

In the tree-view, click the Variables node to see the variables and parameters of the top of the model.

If you click the plus sign of the Variables node, the tree expands and shows the underlying submodels. The plus sign changes into a minus sign. if you click on the minus sign, the tree collapses again.

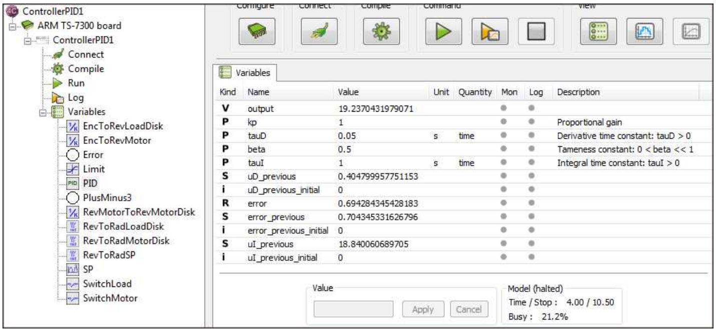


Kind	Name	Value	Unit	Quantity	Mon	Log	Description
I	EncoderLoad	0			📈	📈	
O	PWM	0.999			📈	●	
I	EncoderMotor	0			●	📈	

Model (halted)
Time / Stop : 5.21 / 10.50
Busy : 34.5%

If you click on the plus sign, the tree will be expanded showing the underlying submodels.

You can select a submodel, to show the signals it contains. If the submodel contains underlying submodels, a plus sign will be shown.



Kind	Name	Value	Unit	Quantity	Mon	Log	Description
V	output	19.2370431979071			●	●	
P	kp	1			●	●	Proportional gain
P	tauD	0.05	s	time	●	●	Derivative time constant: tauD > 0
P	beta	0.5			●	●	Tameness constant: 0 < beta << 1
P	tauI	1	s	time	●	●	Integral time constant: tauI > 0
S	uD_previous	0.404799957751153			●	●	
i	uD_previous_initial	0			●	●	
R	error	0.694284345428183			●	●	
S	error_previous	0.704345331626796			●	●	
i	error_previous_initial	0			●	●	
S	uI_previous	18.840060689705			●	●	
i	uI_previous_initial	0			●	●	

Value

Model (halted)
Time / Stop : 4.00 / 10.50
Busy : 21.2%

You can select a submodel to show its variables and show and change its parameters.

You can click the plus sign to expand and show all the submodels. The plus sign will change into a minus sign etc.

5.8 Monitoring and Logging

When a model is running on a target, you can inspect the values of all the variables to see if it is working as it should or to perform measurements. 20-sim 4C provides two ways of inspecting variables: Monitoring and Logging.

- **Monitoring**: inspect variables at real-time when running.
- **Logging**: inspect variables off-line.

Monitoring is useful for a quick inspection or to measure slow varying variables. Because the variable values have to be transferred from the target to 20-sim 4C during runtime without disturbing the running model, the refresh rate for monitoring is limited. This means once every few milliseconds a new value will be shown and the values in between are missed. Monitoring is also referred to as signal tracing.

Logging is useful for detailed measurements of all variables. During logging, the values are stored in the target memory. When the logging is stopped or finished, the results are transferred from the target to 20-sim 4C and stored in a file. Logging is also referred to as data acquisition.

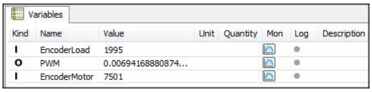
5.9 Run with Monitoring

After configuring, connecting and compiling, you can run a model on the target. The variables (and parameters...) can be monitored in three ways:

- Monitoring in the **variables view**
- Monitoring in a **real-time plot**
- Monitoring in an **oscilloscope view**

Monitoring in the Variables view

When running the model on the target, all variables can be monitored (inspected) by opening a model node under Variables in the tree. For each model node in the tree the variable values are displayed as they change on the running target. Note that monitoring may not show all the samples since a limited refresh rate is used to prevent disturbing the running model.



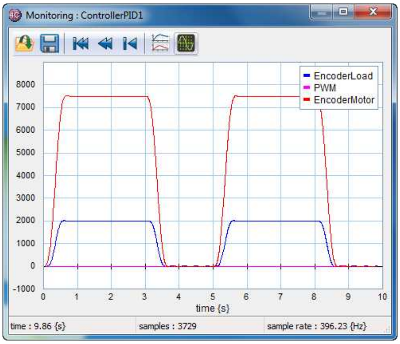
Variables

Kind	Name	Value	Unit	Quantity	Mon	Log	Description
I	EncoderLoad	1995				●	
O	PWM	0.00694168880874...				●	
I	EncoderMotor	7501				●	

Variable inspection in the Variables view during run-time.

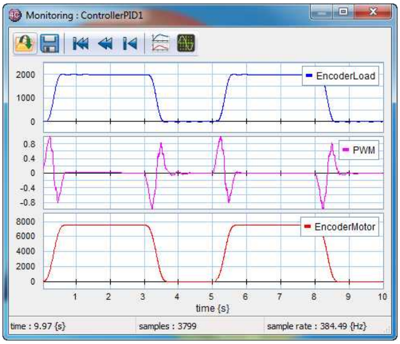
Monitoring in a real-time plot

The second option is to view monitored values in a real-time plot (a live plot). First variables need to be selected for monitoring, see Select variables for logging and monitoring for more details. After variables have been selected, press the *Monitor* button on the action bar at the top of the 20-sim 4C windows. This will open real-time monitoring plot. The *Monitor* button can be pressed before the model is running and during run-time. When the model is already running, the monitor plot shows the variables from starting from the time you opened the monitoring plot. When the monitor plot is already open before running, it will plot the variables from the start. The picture below shows a sample of the real-time plot.



Click the Monitor button to open a plot showing the monitored variables.

By default, 20-sim 4C shows all monitor signals in one plot with a common Y-axis. This is not always convenient, e.g. when the scale of the monitored variables is different.



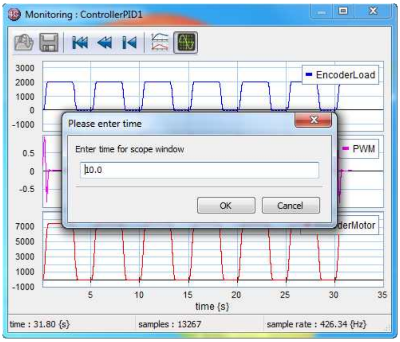
time : 9.97 {s}   samples : 3799   sample rate : 384.49 {Hz}

The same monitor plot with distributed curves.

You can use the Distribute curves button on the toolbar to give each monitored variable its own Y-axis:

Monitoring in an oscilloscope view

The monitor plot shows a signal from the start of the monitoring until the end of the model run. It is also possible to show only the recent history of the variables similar to an oscilloscope. Click on the Enable scope button to switch to the oscilloscope view. Enter the time base for the scope window, e.g. 10 s.



Please enter time

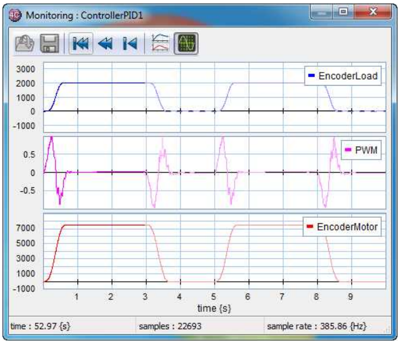
Enter time for scope window

10.0

time : 31.80 {s}   samples : 13267   sample rate : 426.34 {Hz}

Enter time for scope window.

20-sim 4C will now only show a signal trace for a period of 10 s. The previous two traces are still visible but blended into the background.



Monitoring : ControllerPID1

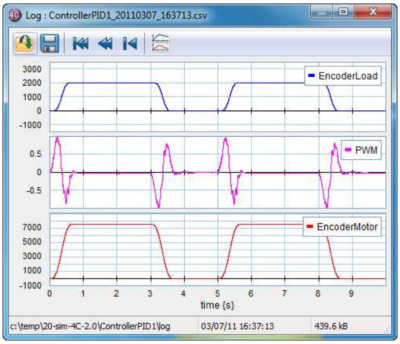
time : 52.97 {s} samples : 22693 sample rate : 385.86 {Hz}

Monitor plot in oscilloscope view.

Use one of the *Clear* buttons to clear older plots.

5.10 Run with Logging

The second button of the Command buttons is *Run with Logging* button . If the button is green the logging has been configured and can be started. Pressing the button will start the model on the target with logging. After the period of logging (which can be configured) has passed, the logging data is transferred from the target to 20-sim 4C and saved to a file. The filename will be shown in the output window. Furthermore, the results will be shown in a plot that will be popped-up after the file has been stored. If the model is still running after logging, it can be stopped by pressing the stop button.



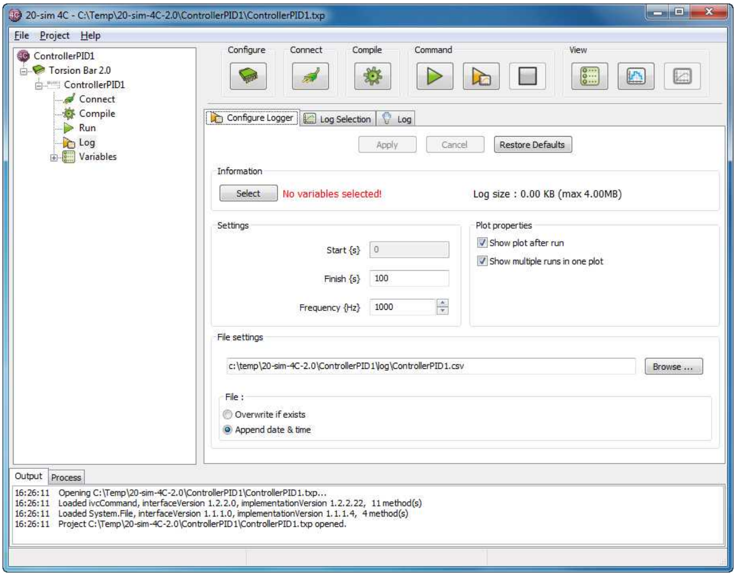
Log : ControllerPID1_20110307_163713.csv

c:\temp\20-sim-4C-2.0\ControllerPID1\log 03/07/11 16:37:13 439.6 kB

After logging has been completed the results can be shown in a plot.

Configure Logger

If the Run with Logging is orange , the logging needs to be configured. Pressing the Run with Logging button now will open the *Configure Logger* tab.



Information

No variables selected!

Log size : 0.00 KB (max 4.00MB)

Settings

Start {s} 0

Finish {s} 100

Frequency {Hz} 1000

Plot properties

☑ Show plot after run

☑ Show multiple runs in one plot

File settings

c:\temp\20-sim-4C-2.0\ControllerPID1\log\ControllerPID1.csv

File :

○ Overwrite if exists

● Append date & time

```
16:26:11   Opening C:\Temp\20-sim-4C-2.0\ControllerPID1\ControllerPID1.txp...
16:26:11   Loaded ivcCommand, interfaceVersion 1.2.2.0, implementationVersion 1.2.2.22,  11 method(s)
16:26:11   Loaded System.File, interfaceVersion 1.1.1.0, implementationVersion 1.1.1.4,  4 method(s)
16:26:11   Project C:\Temp\20-sim-4C-2.0\ControllerPID1\ControllerPID1.txp opened.
```

In the Configure Logger tab you can select the log file.

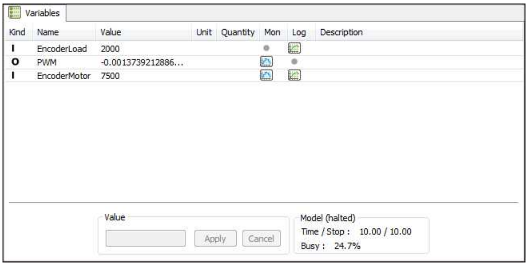
The following options can be configured:

- **Information**: The information box shows how many variables are selected for logging and an estimation of the required (target) memory for logging. The *Select* button will open the Log Selection tab where variables can be selected for logging by clicking the dot in the Log column. See Select variables for logging and monitoring for more details. When no variables are selected, 20-sim 4C will show a red warning message next to the *Select* button and the logging is not configured completely. (It makes no sense to start the logging process without variables to log.)
- **Settings**: The finish time and sample frequency can be specified here. By default, 20-sim 4C will log every sample and for the full run-time. You can lower the frequency to log, for example 1 of every 5 samples. In this way, you can log more variables simultaneously with similar memory requirements.
- **File settings**: The filename and the directory can be specified by pressing either the *Browse* button or type the path and filename in the text field. Furthermore you can choose to overwrite existing files or to add the date & time to the filename.
- **Plot properties**: A plot will be shown after the logging if the Show plot after run option is selected. Multiple runs will be shown in one plot if the Show multiple runs in one plot option is selected.

You have to click the Apply button after the settings have been entered. The *Run with Logging* button will then change from orange into green. If an error occurs the *Run with Logging* button will turn into red.

5.11 Select variables for monitoring and logging

Inputs, outputs and internal variables (and parameters...) can be selected for monitoring and/or logging. You can use the tree-view by selecting the Variables node. If you click the plus size, to expand you will see all the variables and parameters that are available in a model on the top-level. If the model contains hierarchy, you can select one of the underlying submodels to select an internal variable for monitoring and/or logging. A variable can be selected for monitoring and/or logging by pressing the dot in respectively the *Mon* column and/or *Log* column. The following picture shows the result.

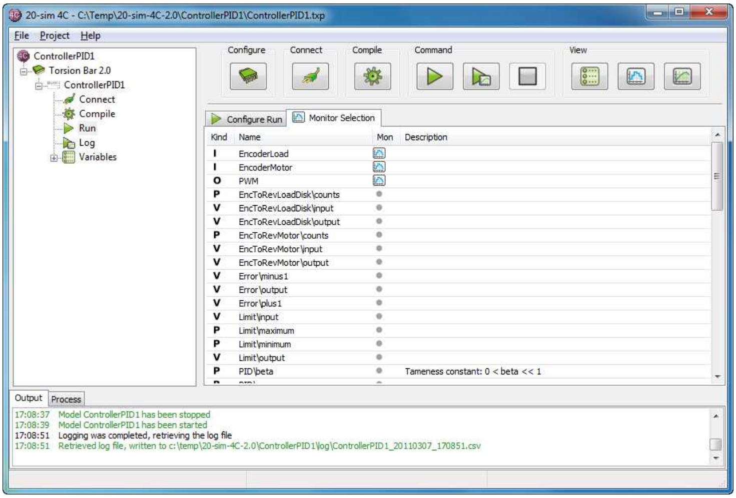


Kind	Name	Value	Unit	Quantity	Mon	Log	Description
I	EncoderLoad	2000			●	📈	
O	PWM	-0.0013739212886...			📈	●	
I	EncoderMotor	7500			📈	📈	

Model (halted)
Time / Stop : 10.00 / 10.00
Busy : 24.7%

Select the variable that should be monitored or logged.

To view the list of all signals with monitored signals select the Run node in the tree followed by clicking the Monitor Selection tab. The following pictures shows an example:

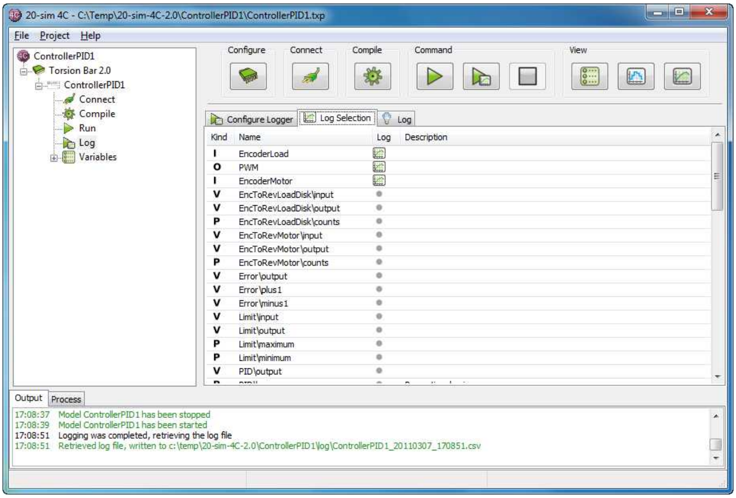


Kind	Name	Mon	Description
I	EncoderLoad	📈	
I	EncoderMotor	📈	
O	PWM	📈	
P	EncToRevLoadDisk\counts	●	
V	EncToRevLoadDisk\input	●	
V	EncToRevLoadDisk\output	●	
P	EncToRevMotor\counts	●	
V	EncToRevMotor\input	●	
V	EncToRevMotor\output	●	
V	Error\minus1	●	
V	Error\output	●	
V	Error\plus1	●	
V	Limit\input	●	
P	Limit\maximum	●	
P	Limit\minimum	●	
V	Limit\output	●	
P	PID\beta	●	Tameness constant: 0 < beta << 1

```
17:08:37   Model ControllerPID1 has been stopped
17:08:39   Model ControllerPID1 has been started
17:08:51   Logging was completed, retrieving the log file
17:08:51   Retrieved log file, written to c:\temp\20-sim-4C-2.0\ControllerPID1\log\ControllerPID1_20110307_170851.csv
```

List of all monitored variables.

To view the list of all signals with logged signals select the Log node in the tree followed by clicking the Log Selection tab. The following picture shows an example.



Kind	Name	Log	Description
I	EncoderLoad	📈	
O	PWM	📈	
I	EncoderMotor	📈	
V	EncToRevLoadDisk\input	●	
V	EncToRevLoadDisk\output	●	
P	EncToRevLoadDisk\counts	●	
V	EncToRevMotor\input	●	
V	EncToRevMotor\output	●	
P	EncToRevMotor\counts	●	
V	Error\output	●	
V	Error\plus1	●	
V	Error\minus1	●	
V	Limit\input	●	
V	Limit\output	●	
P	Limit\maximum	●	
P	Limit\minimum	●	
V	PID\output	●	

```
17:08:37   Model ControllerPID1 has been stopped
17:08:39   Model ControllerPID1 has been started
17:08:51   Logging was completed, retrieving the log file
17:08:51   Retrieved log file, written to c:\temp\20-sim-4C-2.0\ControllerPID1\log\ControllerPID1_20110307_170851.csv
```

List of all logged variables.

5.12 Change Parameters

You can control the behaviour of a running model by changing parameter values on-line:

1. In the tree-view click the ***Variables*** node.

Now you will see all the variables and parameters that are available in a model. If the model contains hierarchy, you can expand it to show all the underlying submodels.

2. Click on the **plus sign** to show all the underlying submodels.

3. Select the desired **submodel**.

4. Select the desired **parameter**.

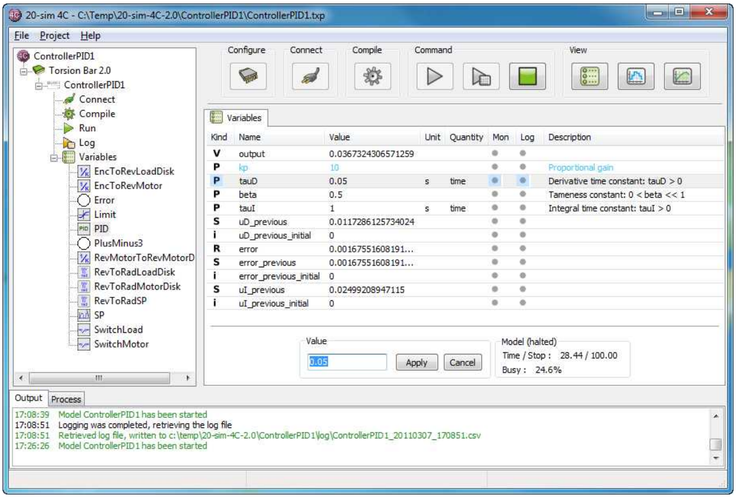
5. In the **Value** field at the bottom of the tab, change the parameter value, followed by the **Enter** key.

The changed value will now become blue to indicate that the value has been changed locally but not yet applied on the running model. The value has not been applied instantly since you may desire to change multiple values at the same time.

6. If desired, repeat the previous steps to change **other parameter values**.

7. Click on the **Apply** button at the bottom of the tab, to apply the changes.

The color of the changed parameters returns to black again when their new values are applied.



Kind	Name	Value	Unit	Quantity	Mon	Log	Description
V	output	0.0367324306571259			●	●	
P	kp	10			●	●	Proportional gain
P	tauD	0.05	s	time	●	●	Derivative time constant: tauD > 0
P	beta	0.5			●	●	Tameness constant: 0 < beta << 1
P	tauI	1	s	time	●	●	Integral time constant: tauI > 0
S	uD_previous	0.0117286125734024			●	●	
i	uD_previous_initial	0			●	●	
R	error	0.00167551608191...			●	●	
S	error_previous	0.00167551608191...			●	●	
i	error_previous_initial	0			●	●	
S	uI_previous	0.02499208947115			●	●	
i	uI_previous_initial	0			●	●	

Value: 0.05 [Apply] [Cancel]

Model (halted)
Time / Stop : 28.44 / 100.00
Busy : 24.6%

```
17:08:39   Model ControllerPID1 has been started
17:08:51   Logging was completed, retrieving the log file
17:08:51   Retrieved log file, written to c:\temp\20-sim-4C-2.0\ControllerPID1\log\ControllerPID1_20110307_170851.csv
17:26:26   Model ControllerPID1 has been started
```

You can change multiple parameters values and click Apply to instantly apply the changes.

5.13 Loading Models

20-sim 4C is automatically started when you generate C-code in 20-sim. You can however start 20-sim 4C manually to open existing 20-sim 4C projects or to create a new project with your own models.

Open an existing 20-sim 4C project

1. From the **Windows Start menu** open **20-sim 4C**.
2. From the **File** menu select **Open Project**. (A model can only be added to a target in a project.)

You will be asked for a name and location of the project. Alternatively, you can double click on a 20-sim 4C project file in the Windows Explorer (Files with .txp extension).

Create a new 20-sim 4C project

1. From the **Windows Start menu** open 20-sim 4C.
2. From the **File** menu select **New Project**. (A model can only be added to a target in a project.)

You will be asked for a name and location of the project.

3. From the **Project** menu select **Add Target.**
4. From the **Project** menu select **Add Model.**

You will be asked for a model name with the extension .mcf which stands for model specification file. This is an XML based file that 20-sim 4C uses for a description of the model.

5.14 Multiple Targets

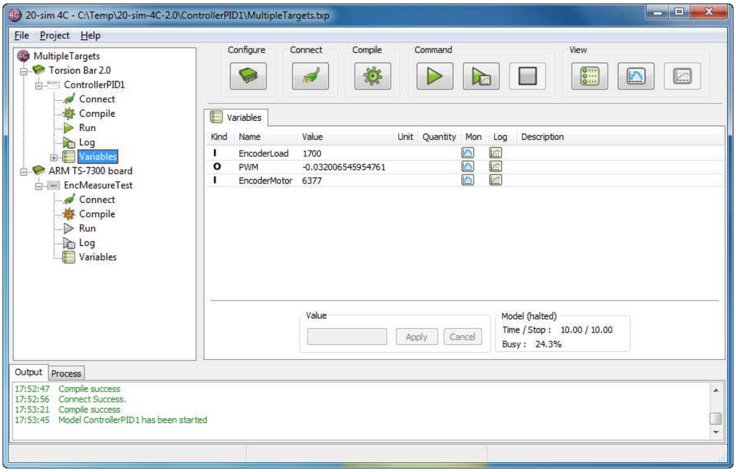
You can control multiple targets from a single version of 20-sim 4C. The easiest way to make this work is to generate a project for each single target and test it separately. Then you create a new project an load the various models:

1. Start **20-sim 4C**.
2. From the **File** menu select **New Project***.*

You will be asked for a name and location of the project.

3. From the **Project** menu select **Add Target**. Choose the first target.
4. From the **Project** menu select **Add Model**. Select the model that corresponds to the first target.
5. Repeat step 3 and 4 for the remaining targets.

Now you are ready to manage all the targets and control their operation from a single instance of 20-sim 4C.



Kind	Name	Value	Unit	Quantity	Mon	Log	Description
I	EncoderLoad	1700					
O	PWM	-0.032006545954761					
I	EncoderMotor	6377					

Model (halted)
Time / Stop : 10.00 / 10.00
Busy : 24.3%

```
17:52:47   Compile success
17:52:56   Connect Success.
17:53:21   Compile success
17:53:45   Model ControllerPID1 has been started
```

A 20-sim 4C project containing 2 targets with different models.

6 Sources

6.1 Introduction

20-sim 4C will accept C/C++ code from various sources:

1. Handwritten C/C++ code

2. 20-sim Professional

3. Scilab : It is currently experimental and only works with RTAI targets.

4. Matlab :It is currently experimental and only works with RTAI targets.

This chapter describes how to use these sources to create C/C++ code for 20-sim 4C.

6.2 Handwritten C-code

If you want to use handwritten C-code as input for 20-sim 4C, please contact Controllab Products.

6.3 20-sim

Requirements

20-sim 4C 2.1 will accept generated C-code from 20-sim Professional version 4.1.2.3 or higher. You can check your the version number of 20-sim in the About Box:

1. Open 20-sim and from the **Help** menu click **About**.

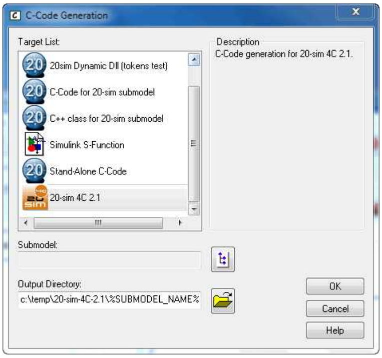
If your version number is lower than 20-sim 4.1.2.3, please update 20-sim first.

Generate C-code in 20-sim

In 20-sim you can generate C-code from any submodel:

1. Build your model and run a **simulation**, until you are satisfied with the response.
2. In the Simulator, from the Tools menu select **Real Time Toolbox** and **C-code Generation**.

The C-code Generation dialog will pop-up.



Description: C-Code generation for 20-sim 4C 2.1.

Output Directory: c:\temp\20-sim-4C-2.1\%SUBMODEL_NAME%

The C-code generation dialog in 20-sim.

If the 20-sim 4C target is not shown, please check if you are using the right version of 20-sim (4.1.2.3 or higher).

3. Select the **20-sim 4C Target**.
4. Select the desired **Submodel**.
5. Click **OK**.

Now 20-sim 4C will be opened with the C-code from 20-sim.

6.4 Scilab

Requirements

- Scilab 5.4.0 Beta 3 - 32 bits or higher
- 20-sim 4C 2.1 or higher

Note: The installation path of 20-sim 4C (e.g. C:\Program Files\20-sim 4C 2.1) will be referred to as "the 20-sim 4C installation dir" below.

Install 20-sim 4C atoms package

1. Install the Atoms-package that is shipped with 20-sim 4C: Start Scilab and run:

```
-> atomsInstall('<20-sim 4C installation dir>/contrib/scilab/20sim_4C_codegeneration-1.0.0.zip')
```

2. Restart Scilab.

Note: Sometimes, the installation script raises an error, in that case, Scilab needs to be restarted one more time.

Generate C-code in Scilab

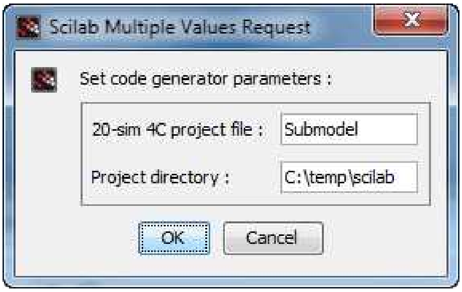
In Scilab you can generate C-code from a sub-model:

1. Open a scilab model for code generation. A sample model found at '<20-sim 4C installation dir>\examples\scilab\demo.xcos' can be used for testing.

2. Select the desired sub-model from the scilab model.

3. Start the code generation from the menu bar > tools > 20-sim 4C code generation.

This will open a dialog on where to save the sub-model.



Set code generator parameters :

20-sim 4C project file : Submodel

Project directory : C:\temp\scilab

A project file saving dialog in Scilab

3. Select a directory and Click OK

This starts the code generation process and 20-sim 4C will be opened.

4. Select your target (*Note: Scilab code generation currently works with RTAI targets only*) and Run the code.

Event triggered Xcos blocks

So far, the event-time triggering the square source is not exported from the Xcos model to 20-sim 4C. Therefore, add the proper event duration to the square block named *gensqr*.

Continous time solvers

In the 20-sim 4C variables section a variable 'solver type' can be found, which can be used to select the desired continuous time solver method by the following values:

- 1 results in ode1
- 2 results in ode2
- 4 results in ode4

Known issues

- The correct event time of external activated blocks within Xcos should be entered manually into 20-sim 4C.
- Block names from within Xcos are not transferred to 20-sim 4C.
- The first model calculation is executed at model initialization. This causes, for example, a square wave to start with the opposite sign.
- The superblock used for code generation cannot itself contain superblocks.

Library blocks not supported

- Continuous time systems
  - CLINDUMMY_f

  - TCLSS
  - TIME_DELAY. Note that the discrete time delay z-1 can be used.
  - VARIABLE_DELAY
  - PDE
- Discontinuities
  - DELAYV_f. This block is not supported because it generates events.
  - RATELIMITER.
  - QUANT_f. The quantizer is not supported yet.
- Discrete time systems
  - AUTOMAT. This block is not supported because it generates events.
  - DELAYV_f. This block is not supported because it generates events.
  - DELAY_f. Note that a delay can be realized by using the z-1 block.
  - DLRADAPT_f. This block is not supported yet.
  - SAMPHOLD_m. Note that in general a sample and hold is used for an analog signal input
  - TCLSS
  - REGISTER
- Lookup tables
  - This palette is not supported yet.
- Event handling
  - no blocks are supported. Note that every clock-triggered block has its own clock within 20-sim 4C. (as long as the desired activation time is >= stepsize of the complete model.
- Mathematical Operations
  - MATMAGPHI. This block is not supported yet.
  - MATZREIM. This block is not supported yet.
  - MAXMIN. This block is not supported yet.
  - MAX_f. This block is not supported yet.
  - MIN_f. This block is not supported yet.
  - POWBLK_f
- Matrix
  - Matrices are not supported yet.
- Electrical
  - no blocks are supported
- Integer
  - No blocks are supported
- Zero crossing detection
  - no blocks are supported
- Signal Routing

  - so far: no blocks are supported
- Signal Processing
  - Not supported. Note that the functions within this palette are in particular useful for the interface between an analog and a digital system.
- Implicit
  - no blocks are supported
- Annotations
  - TEXT_f annotations are not taken into account for the code generation
- Sinks
  - no blocks are supported
- Sources
  - FROMWSB
  - RAND_m
  - READAU_f
  - READC_f
  - RFILE_f
  - Sigbuilder. This block is not supported because it generates events.
- Thermo-Hydraulics
  - no blocks are supported
- Demonstrations Blocks
  - no blocks are supported
- User-Defined Functions
  - no blocks are supported

6.5 Matlab

Requirements

- Matlab R2011b
- Simulink Coder Version 8.1
- 20-sim 4C 2.1 or higher
- Note: The installation path of 20-sim 4C (e.g. C:\Program Files\20-sim 4C 2.1) will in this manual be referred to as the 20-sim 4C installation directory.

Install Real-Time Workshop 20-sim target

1. Open Matlab and go to '<20-sim 4C installation directory>\contrib\matlab \rtw_20sim4C_target\rtw_20sim4C_target'

2. Add the 20-sim target to Matlab: execute ' rtw_20sim4C_target_setup' in the command line using:

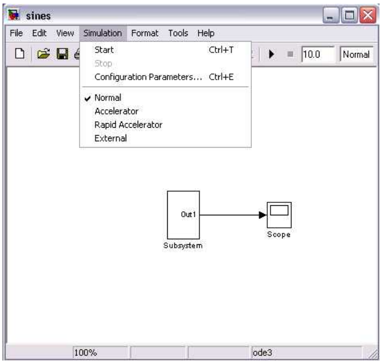
```
run('rtw_20sim4C_target_setup')
```

Test Model

1. In Matlab, go to the 20-sim 4C installation directory and then open the 'examples/ matlab' directory.
   (For the code generation, the simulink model file should be within the present working directory)

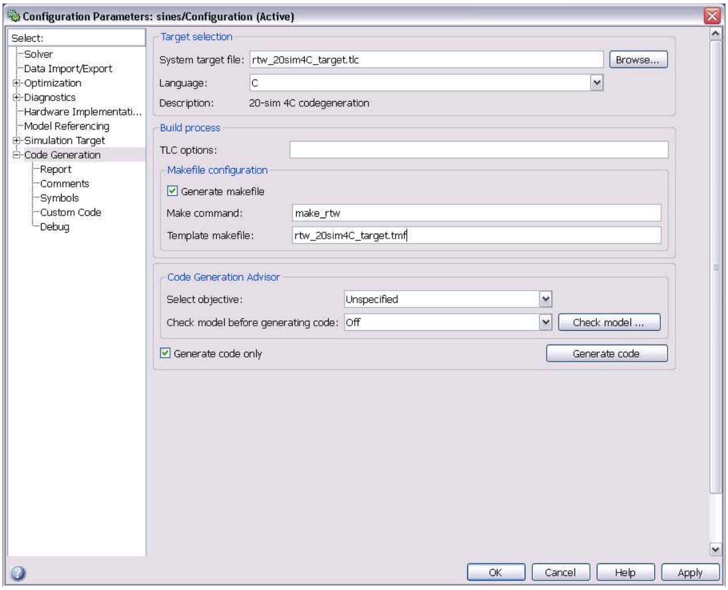
2. Open the model 'sines.mdl'.

3. Go to Simulation -> Configuration Parameters:



Configuration parameters

This will open the *configuration parameters dialog* shown below



Target selection

System target file: rtw_20sim4C_target.tlc

Language: C

Description: 20-sim 4C codegeneration

Build process

TLC options:

Makefile configuration

☑ Generate makefile

Make command: make_rtw

Template makefile: rtw_20sim4C_target.tmf

Code Generation Advisor

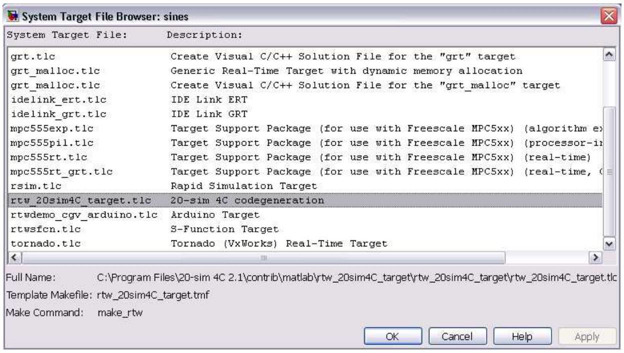
Select objective: Unspecified

Check model before generating code: Off

☑ Generate code only

Configuration Parameters dialog

Make sure the rtw_20sim4C_target.tlc is selected in the "system target file" section. If not, Click 'Browse' to select the correct file.



System Target File:	Description:
grt.tlc	Create Visual C/C++ Solution File for the "grt" target
grt_malloc.tlc	Generic Real-Time Target with dynamic memory allocation
grt_malloc.tlc	Create Visual C/C++ Solution File for the "grt_malloc" target
idelink_ert.tlc	IDE Link ERT
idelink_grt.tlc	IDE Link GRT
mpc555exp.tlc	Target Support Package (for use with Freescale MPC5xx) (algorithm e:
mpc555pil.tlc	Target Support Package (for use with Freescale MPC5xx) (processor-i:
mpc555rt.tlc	Target Support Package (for use with Freescale MPC5xx) (real-time)
mpc555rt_grt.tlc	Target Support Package (for use with Freescale MPC5xx) (real-time, (
rsim.tlc	Rapid Simulation Target
rtw_20sim4C_target.tlc	20-sim 4C codegeneration
rtwdemo_cgv_arduino.tlc	Arduino Target
rtwsfcn.tlc	S-Function Target
tornado.tlc	Tornado (VxWorks) Real-Time Target

Full Name: C:\Program Files\20-sim 4C 2.1\contrib\matlab\rtw_20sim4C_target\rtw_20sim4C_target\rtw_20sim4C_target.tlc

Template Makefile: rtw_20sim4C_target.tmf

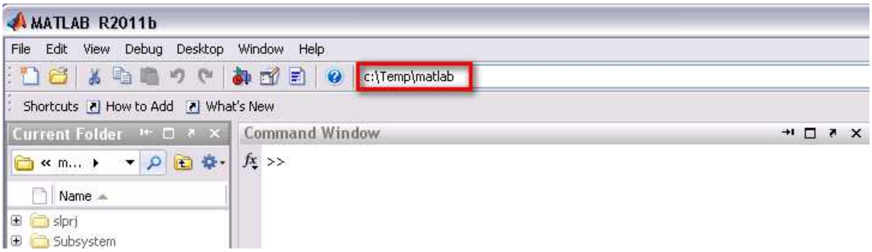
Make Command: make_rtw

System target file selection

4. Click OK to close the *configuration parameters dialog.*

5. Select the sub model "*Subsystem"* you need to generate code for.

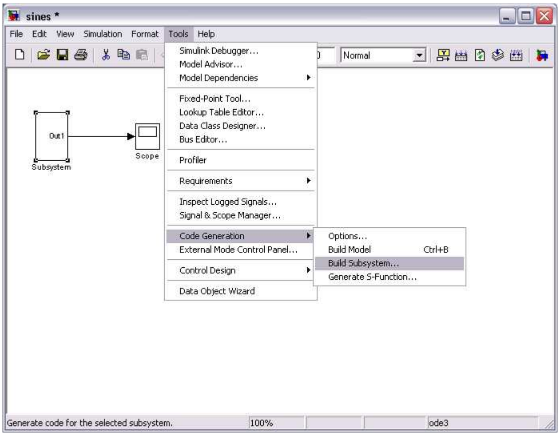
6. Go to the editor window and change the current directory to a temporary working directory as shown below.


MATLAB R2011b
File Edit View Debug Desktop Window Help
Shortcuts How to Add What's New
Current Folder
Command Window
« m...
Name
slprj
Subsystem
fx >>

c:\Temp\matlab

Choosing temporary working directory for saving project files

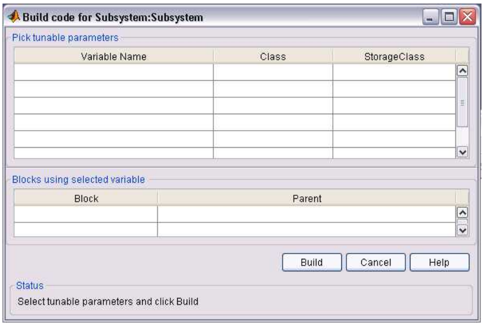
6. Go to Tools -> Code Generation -> Build Subsystem. Make sure a sub-model is selected first, otherwise the 'Build Subsystem' option remains gray.



Generate code for the selected subsystem.

Building selected subsystem

Selecting 'Build Subsystem' should open the window shown below.



Select tunable parameters and click Build

"Build" button will open the 20sim4C window.

7 Targets

7.1 Introduction

You can choose from a set of built-in targets in 20-sim 4C 2.1 or define your own targets. The targets should be equipped with a real-time operating system that is supported by 20-sim 4C.

Built-in targets

1. Bachmann M1 Controller: Industrial PC-based hardware and I/O modules designed to withstand the toughest ambient conditions.
2. TS 7300: This is the TS-7300 ARM 9 board of Technologic Systems running RTAI Linux optionally equipped with a TS-9700 or TS-ADC16 AD/DA I/O board.
3. TS 7300 CUSTOM : This is the TS-7300 board with a custom FPGA configuration enabling 5xPWM output and 5XEncoder input.
4. Torsion Bar v1.0: This a demonstration setup based on the TS-7300 equipped with a PWM motor amplifier.
5. Torsion Bar v2.0: This a demonstration setup based on the TS-7300 equipped with a current motor amplifier.
6. Generic PC or industrial PC: This is a generic PC / PC/104 target without defined I/O running RTAI Linux or Xenomai.
7. PC/PC104 with Sensoray 526: This is a generic PC/PC104 target with the Sensoray 526 multifunction IO board (www.sensoray.com) running RTAI Linux.

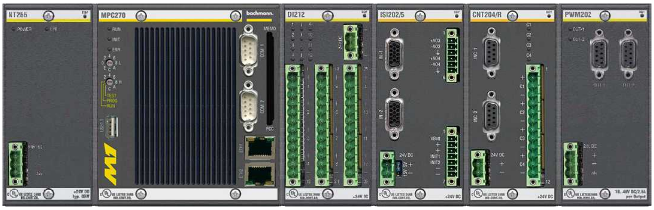
Define your own targets

Each target is defined by a so called *Target Configuration file*. By defining a new target Configuration file (extension .tcf) a new target will be defined. The format is open and you can define your own targets if you like to do so. Please contact Controllab Products B.V. for more details about creating/defining your own targets.

Note*: Although the tcf format is backward compatible, tcf from 20-sim 4C 2.0 may not work in 20-sim 4C 2.1. Therefore, an existing 20-sim 4C 2.0 .tcf file needs an upgrade to 2.1.*

7.2 Bachmann M1

The *Bachmann M1* controller perfectly combines the openness of a PC-based controller with the reliability of industrial hardware platforms. Designed to withstand the toughest ambient conditions it guarantees error-free use over long periods of time, at subzero temperatures and without fans. A modern system architecture designed for consistent network-capability permits the easy integration of the M1 into the environment of the controller and system peripherals. Real-time ethernet permits the real-time networking of the controllers, and the support of all standard Fieldbus systems permits the connection of standard external components.



Bachmann M1 Controller with I/O modules.

Supported Controllers and I/O modules

Bachmann provides a broad range of powerful CPU's with a broad selection of I/O modules. The following I/O modules are currently supported :

- CNT204 - Counter Module
- ACR222/2 - Stepper Motor Module
- ISI222/8 - Positioning Module (encoder + analog out)
- PWM202 - Pulse Width Modulation Module
- AI204 - Analog Input Module
- AIO288 - Analog Input/Output Module
- AO202 - Analog Output Module
- AO208 - Analog Output Module
- DIO216 - Digital Input/Output Module
- DIO232 - Digital Input/Output Module
- DI216 - Digital Input Module
- DI232 - Digital Input Module
- DO216 - Digital Output Module
- DO232 - Digital Output Module

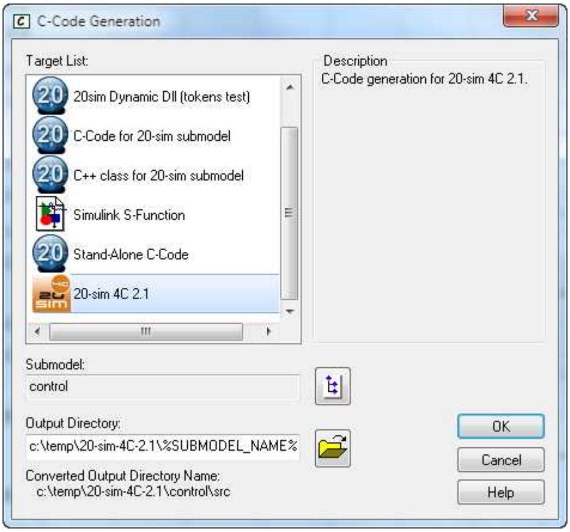
If your module is unsupported please contact Controllab Products B.V. for more information.

Supplier

The Bachmann M1 Controller is a product of Bachmann Electronic GmbH. More information can be found on the website: http://www.bachmann.info

Configuration for 20-sim

In order to use the Bachmann target from 20-sim, choose the 20-sim 4C C-Code target.



Description: C-Code generation for 20-sim 4C 2.1.

Submodel: control

Output Directory: c:\temp\20-sim-4C-2.1\%SUBMODEL_NAME%

Converted Output Directory Name: c:\temp\20-sim-4C-2.1\control\src

20-sim 4C C-Code target

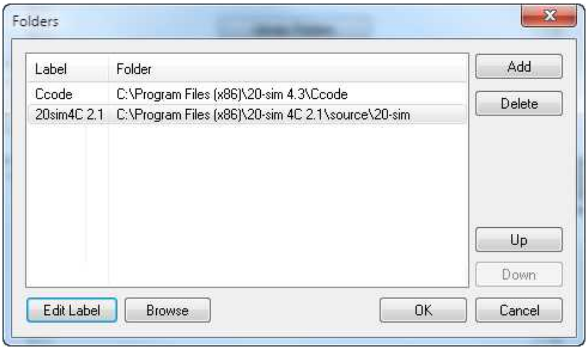
If the "20-sim 4C 2.1" target is not listed, the target needs to be added.
Open 20-sim and go to the following menu: Tools \ Options \ Folder \ C-Code Folders
Add the C-Code generation template path to the C-code Folders.

Default location Windows (32-bit):

```
C:\Program Files\20-sim 4C 2.1\source\20-sim
```

Default location Windows (64-bit):

```
C:\Program Files (x86)\20-sim 4C 2.1\source\20-sim
```



Label	Folder
Ccode	C:\Program Files (x86)\20-sim 4.3\Ccode
20sim4C 2.1	C:\Program Files (x86)\20-sim 4C 2.1\source\20-sim

Adding 20-sim 4C target

Time base

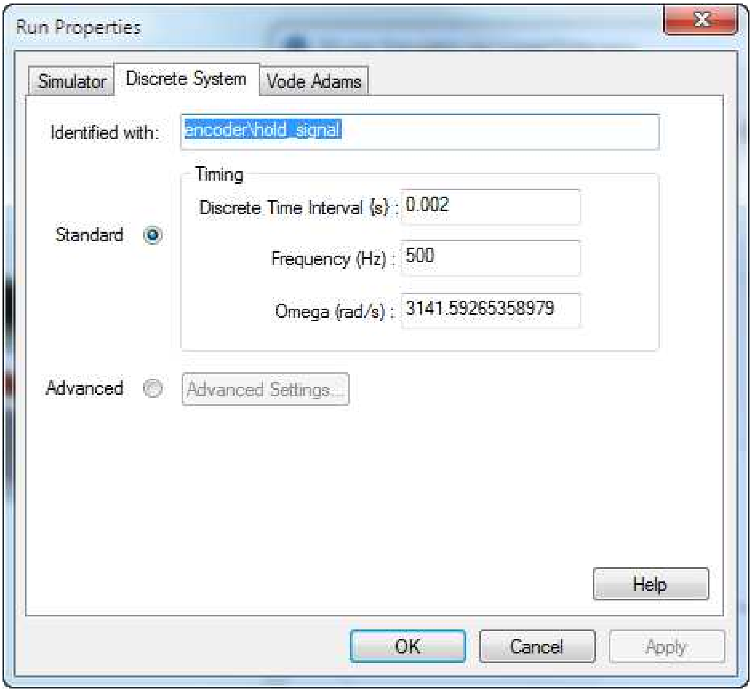
In order to set the time base for your application take note of the following guidelines.

- Set the desired step size in the *Run Properties* dialog of 20-sim for simulation.
- The step size can be changed in the *Configure Run* settings in 20-sim 4C.
- Set the *System Time Base* in the *Bachmann SolutionCenter*
    - **Ticks / second** : Set at least as high as the frequency of your application preferably higher. e.g. application frequency is 500 Hz, set the "Ticks / second" to 2000 and "Ticks / Timeslice" to 2, this will amount to 1000Hz per operating system task.
    - **Sync - Signal** : Set both times to the **same** value. The value needs to be an exact multiple of the step size. During application initialization (on the M1 target) the time base is initialized of no extract multiple can be found, the application will be terminated!

As example consider a step size of 0.002 s. This is 2000 us and this is exact multiplier of 4 times 500 us.

If the Sync Signal times are set to 500 us, the following step sizes are possible:
0.0005 s, 2 kHz (sync count = 1)
0.001 s, 1 kHz (sync count = 2)
0.0015 s, 666.667 Hz (sync count = 3)
0.002 s, 500 Hz (sync count = 4)
0.0025 s, 400 Hz (sync count = 5)
and so on...

The following example shows the 20-sim *Run* Properties dialog configuration for an application that runs at 500Hz.



Run Properties

Identified with: encoder\hold_signal

Timing

Standard ●

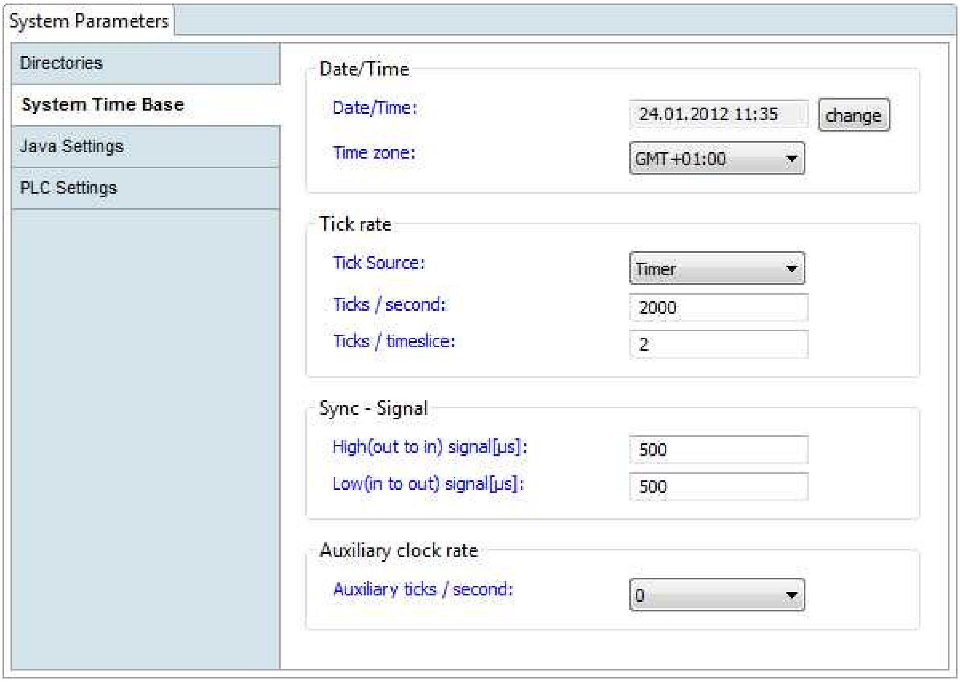
Discrete Time Interval {s} : 0.002

Frequency (Hz) : 500

Omega (rad/s) : 3141.59265358979

Advanced ○

The corresponding System Time Base settings in the Bachmann Solution Center:



Date/Time

Date/Time: 24.01.2012 11:35

Time zone: GMT+01:00

Tick rate

Tick Source: Timer

Ticks / second: 2000

Ticks / timeslice: 2

Sync - Signal

High(out to in) signal[µs]: 500

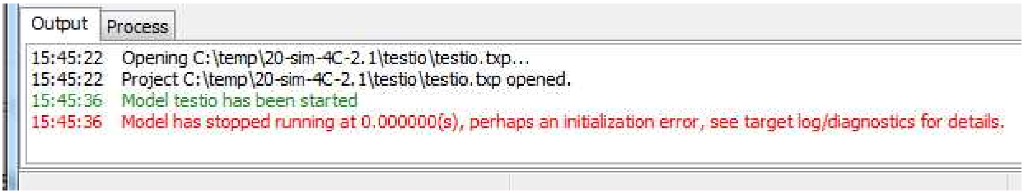
Low(in to out) signal[µs]: 500

Auxiliary clock rate

Auxiliary ticks / second: 0

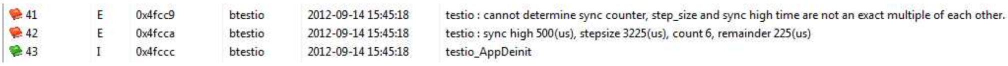
If the step size does not match an exact multiple of the Sync - Signal the application will be terminated.

The following message will appear in 20-sim 4C.



```
15:45:22  Opening C:\temp\20-sim-4C-2.1\testio\testio.txp...
15:45:22  Project C:\temp\20-sim-4C-2.1\testio\testio.txp opened.
15:45:36  Model testio has been started
15:45:36  Model has stopped running at 0.000000(s), perhaps an initialization error, see target log/diagnostics for details.
```

The following message will show in the Diagnostics log of the Solution Center



```
41   E   0x4fcc9   btestio   2012-09-14 15:45:18   testio : cannot determine sync counter, step_size and sync high time are not an exact multiple of each other.
42   E   0x4fcca   btestio   2012-09-14 15:45:18   testio : sync high 500(us), stepsize 3225(us), count 6, remainder 225(us)
43   I   0x4fccc   btestio   2012-09-14 15:45:18   testio_AppDeinit
```

Run application at boot

In order to run the application at boot time on the Bachmann M1 controller add the following lines to mconfig.ini
In this case the application/module name is testio.

Note: the ModuleName is case-sensitive and must match exactly

```
[TESTIO]
  (BaseParms)
    Partition = 2
    DebugMode = 0x0
    Priority = 130
    ModuleIndex = 0
    ModulePath = /cfc0/app/xxsim/
    ModuleName = testio.m
```

FAQ

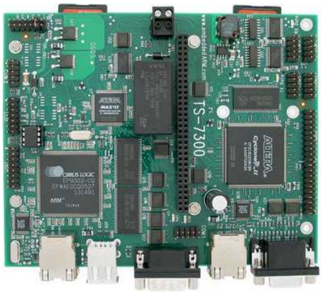
Q) Error that a SVI variable exceeds 63 characters.
A) Set the system compatibility in the mconfig.ini to "Compatibility = 320"

Q) My output is not reset when the model is stopped.
A) Make sure that your Bachmann configuration (tcf) contains a <CLOSE> and <DESTRUCT> tag for proper closing of the component.

7.3 TS-7300

Description

The TS-7300 is a compact Single Board Computer based upon the Cirrus EP9302 ARM9 CPU and set of on-board peripherals. The board is equipped with a SD-card which contains the OS.



The ARM 7300 board.

Inputs and Outputs

- The on-board FPGA DIO2 provides digital input & output.
- The TS_9700 peripheral board provides ADC and DAC ports for data acquisition in analog applications.
- The TS-ADC16 peripheral board provides ADC and DAC ports for data acquisition in analog applications.

Supplier

The TS-7300 is a product of Technologic Systems. More information can be found on the website: http://www.embeddedarm.com/products/board-detail.php?product=TS-7300.

Operating System

A TS-7300 board ordered from Controllab Products B.V. is delivered with the following operating system:

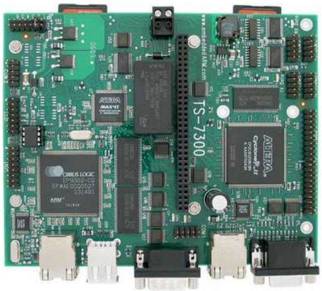
- RTAI real-time Linux including the 20-sim 4C target support software.

You can also order the SD Flash card with our RTAI real-time linux installation including the 20-sim 4C target support software separately from Controllab Products B.V.

7.4 TS-7300 CUSTOM

Description

The TS-7300 is a compact Single Board Computer based upon the Cirrus EP9302 ARM9 CPU and set of on-board peripherals. The board is equipped with a SD-card which contains the OS.
The **TS-7300 CUSTOM** target provides up to **5xPWM** outputs and **5xEncoder** inputs by using an alternative FPGA configuration for the DIO2 connector.



The ARM 7300 board.

Inputs and Outputs

- The CUSTOM FPGA DIO2 provides more digital input & output than the default configuration.
- The TS_9700 peripheral board provides ADC and DAC ports for data acquisition in analog applications.
- The TS-ADC16 peripheral board provides ADC and DAC ports for data acquisition in analog applications.

Supplier

The TS-7300 is a product of Technologic Systems. More information can be found on the website: http://www.embeddedarm.com/products/board-detail.php?product=TS-7300.

Operating System

A TS-7300 board ordered from Controllab Products B.V. is delivered with the following operating system:

- RTAI real-time Linux including the 20-sim 4C target support software.

You can also order the SD Flash card with our RTAI real-time linux installation including the 20-sim 4C target support software separately from Controllab Products B.V.

7.5 TS-ARM I/O

FPGA DIO2 Default

Note: The DIO2 port is only the right 20-pin part of the 40 pins FPGA connector. The first 20-pins are used by the VGA output.

The following digital inputs & output are provided

Digital Inputs

- 2 x Quadrature Encoder Input. The Encoder Input can have no reset and a reset on positive or negative index pulse.
- 2 x Edge Counter Input. The Edge Counter Input can have no reset and a reset on positive or negative edge.
- 8 x Digital Input. Binary input.

Digital Outputs

- 2 X Pulse Width Modulation Output. Pulse Width Modulation output e.g. used to drive a motor via a H-Bridge.
- 8 x Digital Output. Binary output.
- 2 x LED. Two LED's are present on the board.

Connector layout

There is a 40-pin header next to the FPGA.
It is devided into two 20 pin blocks. One is labeled DIO2 and contains the 18 dedicated GPIO pins. The other contains 17 signals that are used by the VGA output but can also be used as GPIO if video is not used.

Warning: All pins on the DIO2 header use 0-3.3V logic levels. Do not drive these lines to 5V.

2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40
1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39
* **VGA** / DIO 3										* DIO2									

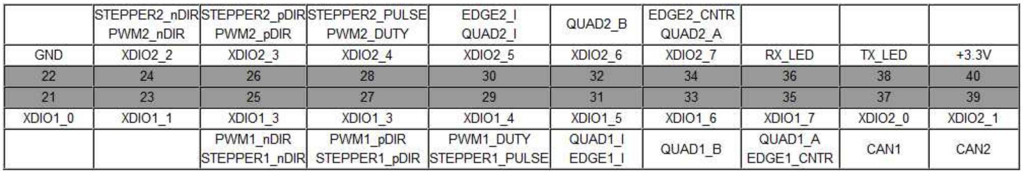
pins #2 and #22 are grounds
pin #20 is fused 5V (polyfuse)
pin #40 is regulated 3.3V
pin #18 can be externally driven high to disable DB15 VGA connector DACs
pin #36 and #38 also go to the red and green LEDs (active low)
pin #39 is a dedicated clock input and cannot be programmed for output

Connector pinout

Default FPGA configuration Controllab

- Contains VGA / TS XDIO core

Note: Pins 1 to 19 are used for the **VGA** output and are not available as digital I/O pins.



	STEPPER2_nDIR PWM2_nDIR	STEPPER2_pDIR PWM2_pDIR	STEPPER2_PULSE PWM2_DUTY	EDGE2_I QUAD2_I	QUAD2_B	EDGE2_CNTR QUAD2_A			
GND	XDIO2_2	XDIO2_3	XDIO2_4	XDIO2_5	XDIO2_6	XDIO2_7	RX_LED	TX_LED	+3.3V
22	24	26	28	30	32	34	36	38	40
21	23	25	27	29	31	33	35	37	39
XDIO1_0	XDIO1_1	XDIO1_3	XDIO1_3	XDIO1_4	XDIO1_5	XDIO1_6	XDIO1_7	XDIO2_0	XDIO2_1
		PWM1_nDIR STEPPER1_nDIR	PWM1_pDIR STEPPER1_pDIR	PWM1_DUTY STEPPER1_PULSE	QUAD1_I EDGE1_I	QUAD1_B	QUAD1_A EDGE1_CNTR	CAN1	CAN2

TS-7300 DIO-2 pin layout

Special function Inputs

Quadrature counter

Quadrature counter input (2x) no index

Quadrature counter with no reset on index pulse. The counter is initialized on 0 and will count to a maximum of 32767 pulses. If maximum number of pulses is exceeded counter will continue to count from -32768.

Quadrature counter input pins: **QUAD?_A and QUAD?_B**

Quadrature counter input (2x) reset in positive index

Quadrature counter with reset on positive flank of the index pulse. The counter is initialized on 0 and will count to a maximum of 32767 pulses. If maximum number of pulses is exceeded counter will continue to count from -32768. If a positive index pulse is received the counter will be reset to zero.Quadrature counter input pins: **QUAD?_A and QUAD?_B**

Index input pin: **QUAD?_I**

Quadrature counter input (2x) reset in negative index

Quadrature counter with reset on negative flank of the index pulse. The counter is initialized on 0 and will count to a maximum of 32767 pulses. If maximum number of pulses is exceeded counter will continue to count from -32768. If a negative index pulse is received the counter will be reset to zero.

Quadrature counter input pins: **QUAD?_A and QUAD?_B**

Index input pin: **QUAD?_I**

Edge counter

Edge counter no index

Edge counter without reset on index pulse. The counter is initialized on 0 and will count to a maximum of 65536 pulses. If maximum number of pulses is exceeded counter will continue to count from 0.

Edge counter input pin: **EDGE?_CNTR**

Edge counter reset on positive edge

Edge counter with reset on negative edge on **EDGE?_I**. Counter is initialized on 0 and will count to a maximum of 65536 pulses. If maximum number of pulses is exceeded counter will continue to count from 0. If a positive edge is received the
counter will be reset to zero.

Edge counter input pin: **EDGE?_CNTR**

Index input pin: **EDGE?_I**

Edge counter reset on negative edge
Edge counter with reset on negative edge on **EDGE?_I**. Counter is initialized on 0 and will count to a maximum of 65536
pulses. If maximum number of pulses is exceeded counter will continue to count from 0. If negative edge is received the
counter will be reset to zero.
Edge counter input pin: **EDGE?_CNTR**
Index input pin: **EDGE?_I**

General purpose digital inputs

Only available when the pins are not used for special function I/O.

16 Digital Inputs
Digital Input Pin: 3.3V is a logical 1, 0V is a logical 0.
Each pin can be individually selected as digital input, digital output
Corresponding value in 20-sim 4C will be either 0.0 or 1.0.
8x on pins: XDIO1_0 ... XDIO1_7
8x on pins: XDIO2_0 ... XDIO2_7

Special function Outputs

PWM
Pulse Width Modulation output (2x)
The frequency is set to 20kHz.
The duty cycle can be specified in 20-sim 4C between [-1.0 , 1.0] (i.e. 0.5 = 50% dutycycle) and is available at pin **PWM?_DUTY**

The sign sets the direction. Positive sign: **PWM?_nDIR** = low and **PWM?_pDIR** = high. Negative sign: **PWM?_nDIR** = high and **PWM?_pDIR** = low.
When the value is zero **PWM?_nDIR** and **PWM?_pDIR** are set to low.

Stepper
Pulse frequency generation output (2x)
The pulse width is fixed to 80 {us}. Therefore the maximum pulse frequency is 6250 {Hz} (=1/160{us}).
The frequency can be specified in 20-sim 4C between [-6250.0, 6250.0] and is available at pin **STEPPER?_PULSE**

The sign sets the direction. Positive sign: **STEPPER?_nDIR** = low and **STEPPER?_pDIR** = high. Negative sign: **STEPPER?_nDIR** = high and **STEPPER?_pDIR** = low.
No pulses are generated when the frequency value is smaller than 5.0 (5 {Hz}) due to limitations in the pulse generator. **STEPPER?, STEPPER?_nDIR and STEPPER?_pDIR** are set to low.

General purpose digital outputs

Only available when the pins are not used for special function I/O.

15 Digital Outputs
Digital Output Pin: A value > 0.0 sets the output to 3.3V and 0.0 sets the output to 0 V
Each pin can be individually selected as digital input, digital output
8x on pins: **XDIO1_0 ... XDIO1_7**
7x on pins: **XDIO2_0, XDIO2_2, XDIO2_3, XDIO2_4, XDIO2_5, XDIO2_6, XDIO2_7**
(XDIO2_1 is not available as output)

FPGA DIO2 Custom

Warning: All pins on the DIO2 header use 0-3.3V logic levels. Do not drive these lines to 5V.

General purpose IO (GPIO/DIO)

Default all pins are accessible as general purpose pins. If a component is enabled on the same pin, the component claims the pin and the pin is no longer available for general purpose usage.

The pins of the 40-pin header mounted at the side of the FPGA can be controlled.

2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40
1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39

Some pins are not available, because the have a special function:

Pin	Function
2	GND
18	OEV (set high by the FPGA, to disable the signals at the VGA connector)
20	5 V
22	GND
40	3.3 V

Available I/O components

1. 35 General purpose digital input / output pins
2. 5x PWM output
3. 5x Frequency generation output (square wave 50% duty) for stepper motor control
4. 5x Encoder input
5. 4x Duty cycle measurement / Frequency measurement / Edge counter input

The 35 general purpose input/output pins are shared with the other components. See the following pin layout table:

Pin layout

This is a layout of the 40-pin connector at the side of the FPGA board

Function 2	Function 1	Pin	Pin	Function 1	Function 2
PWM2 C / Frequency2 C	GPIO 1	1	2	GND	
PWM2 A / Frequency2 A	GPIO 3	3	4	GPIO 4	PWM2 B / Frequency2 B
Count / Quadrature Enc 2 (Index)	GPIO 5	5	6	GPIO 6	Count / Quadrature Enc 2 (A)
Count / Quadrature Enc 2 (B)	GPIO 7	7	8	GPIO 8	Count / Quadrature Enc 4 (A)
PWM4 A / Frequency4 A	GPIO 9	9	10	GPIO 10	Count / Quadrature Enc 4 (B)
PWM4 B / Frequency4 B	GPIO 11	11	12	GPIO 12	Count / Quadrature Enc 4 (Index)
PWM4 C / Frequency4 C	GPIO 13	13	14	GPIO 14	Duty / Frequency / Edge measurem. 3
PWM5 A / Frequency5 A	GPIO 15	15	16	GPIO 16	Duty / Frequency / Edge measurem. 4
PWM5 B / Frequency5 B	GPIO 17	17	18	OEV	
PWM5 C / Frequency5 C	GPIO 19	19	20	5 V	
PWM1 C / Frequency1 C	GPIO 21	21	22	GND	
PWM1 A / Frequency1 A	GPIO 23	23	24	GPIO 24	PWM1 B / Frequency1 B
Count / Quadrature Enc 1 (Index)	GPIO 25	25	26	GPIO 26	Count / Quadrature Enc 1 (A)
Count / Quadrature Enc 1 (B)	GPIO 27	27	28	GPIO 28	Count / Quadrature Enc 3 (A)
PWM3 A / Frequency3 A	GPIO 29	29	30	GPIO 30	Count / Quadrature Enc 3 (B)
PWM3 B / Frequency3 B	GPIO 31	31	32	GPIO 32	Count / Quadrature Enc 3 (Index)
PWM3 C / Frequency3 C	GPIO 33	33	34	GPIO 34	Count / Quadrature Enc 5 (A)
Duty / Frequency / Edge measurem. 1	GPIO 35	35	36	GPIO 36	RED LED / Count / Quadrature Enc 5 (B)
Duty / Frequency / Edge measurem. 2	GPIO 37	37	38	GPIO 38	Green LED / Count / Quadrature Enc 5 (Index)
	GPIO 39 (input only)	39	40	3.3 V	

PWM

The custom-FPGA configuration has 5 PWM outputs.

PWM A = Pulse width modulated signal. Frequency = 16 kHz, Duty cycle setpoint resolution 12 bits
PWM B = Direction signal
PWM C = Break signal

Frequency generator / Stepper motor

The custom-FPGA configuration has 5 Frequency outputs.

Frequency A = Square wave output. Frequency range: 0-3 MHz, resolution 0.05 Hz, duty cycle 49-51%
Frequency B = Direction signal
Frequency C = Break signal

Encoder

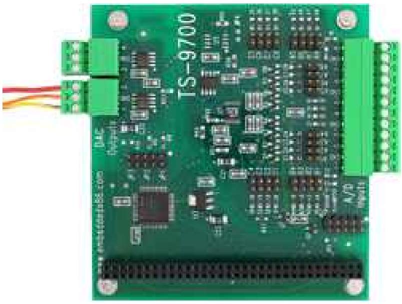
The custom-FPGA configuration has 5 encoder inputs.

Encoder channel A = A signal
Encoder channel B = B signal
Encoder index = index signal

TS-9700

Description

The TS-7300 with TS-9700, is the ARM 7300 mounted with the TS-9700 peripheral board. The TS-9700 provides additional analog Inputs/Outputs.



The TS-9700 peripheral board.

Inputs and Outputs

Additional to the interface of the ARM 7300 the following inputs and outputs are available:

Analog Inputs

- 8 channel 12-bit Analog to Digital Converter with precision 0.2% Analog Reference. Each A/D channel is jumper selectable for: 0 to 2.5V input, 0 to 10V input, 0 to 20mA input.

Analog Outputs

- 4 channel 12-bit Digital to Analog Converter (optional). DAC outputs are 0 to 5V range.

Jumper settings

- **Default settings for 20-sim 4C:**
  - no jumpers: address 0x160, 0-2.5 V input range
- **Input range selection:**
  - 0-2.5 V input range, 120 Hz bandwidth: remove A, B and C jumpers for a particular channel
  - 0-2.5 V input range, 10 kHz bandwidth: place jumper A and remove the B and C jumpers for a particular channel
  - 0-10 V input range, 500 Hz bandwidth: remove the A and B jumper and place only the C jumper for a particular channel
  - 0-20 mA input range: place jumper A and B and remove jumper C for a particular channel

See for more details the official TS-9700 wiki page: http://wiki.embeddedarm.com/wiki/TS-9700

Supplier

The TS 7300 is a product of Technologic Systems. More information can be found on their website: http://www.embeddedarm.com/products/board-detail.php?product=TS-9700 and their TS-9700 wiki page: http://wiki.embeddedarm.com/wiki/TS-9700.

Operating System

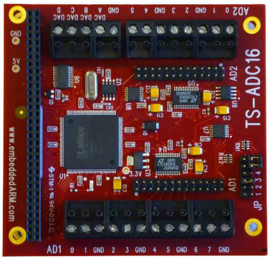
The TS 7300 board is delivered with the following operating system:

- RTAI real-time Linux: You can order an SD Flash card with a customized RTAI real-time linux installation including the 20-sim 4C target support software.

TS-ADC16

Description

The TS-7300 with TS-ADC16, is the ARM 7300 equipped with a TS-ADC16 multi-I/O peripheral board. The TS-9700 provides additional analog Inputs/Outputs, digital inputs, digital counters and a digital output.



The TS-ADC16 peripheral board.

Inputs and Outputs

Additional to the I/O provided on the ARM 7300 the following inputs and outputs are available:

Analog Inputs

- 16 channel 16-bit Analog to Digital Converter. Each A/D channel has the following software programmable input ranges: 0 to +5V, 0 to +10V, -5 to +5V, -10 to +10V input, 0 to 10V input single ended or differential.

Analog Outputs

- 4 channel 12-bit Digital to Analog Converter. DAC outputs are 0 to 5V range.

Digital Inputs

- 4 digital inputs 0-5 V
- 4 16-bit digital counters

Digital Outputs

- 1 digital output - 0 to 3.3V/50mA

Jumpers

Default jumper settings:

- JP1 and JP2 removed (address=0x100)
- JP3 placed (ARM mode)
- JP4 removed (IRQ 6)

Supplier

The TS 7300 is a product of Technologic Systems. More information can be found on the website: http://www.embeddedarm.com/products/board-detail.php?product=TS-ADC16 and their TS-ADC16 wiki page: http://wiki.embeddedarm.com/wiki/TS-ADC16.

Operating System

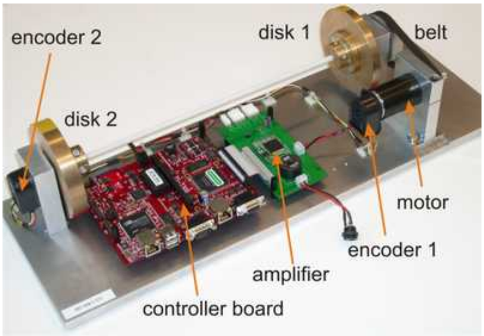
The TS 7300 board is delivered with the following operating system:

- RTAI real-time Linux: You can order an SD Flash card with a customized RTAI real-time linux installation including the 20-sim 4C target support software.

7.6 Torsion Bar v1.0

The Torsion Bar Setup is a compact affordable demonstration setup that demonstrates some important features of a real machine:

- Motor inertia
- Load
- Drive flexibility



The Torsion Bar Setup v1.0.

Inputs and Outputs

The interface of the TS-7300 has been limited to the following inputs and outputs:

Digital Inputs

- 2 x Quadrature Encoder Input. The Encoder Input can have no reset and a reset on positive or negative index pulse.

Digital Outputs

- 1 PWM output, to drive the motor via the H-bridge which is on the interface board.
- 2 x LED. Two LED's are present on the board.

Supplier

The Torsion Bar Setup is a product of Controllab Products B.V. Please contact Controllab Products B.V. for more details about this set-up.

Operating System

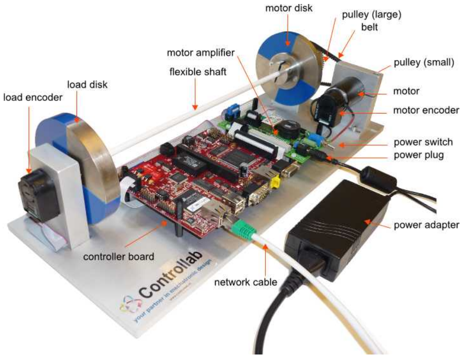
The Torsion Bar v1.0 setup is delivered with the following operating system:

- RTAI real-time Linux: The setup runs our customized RTAI real-time linux installation including the 20-sim 4C target support software.

7.7 Torsion Bar v2.0

The Torsion Bar Setup is a compact demonstration setup that demonstrates some important features of a real machine:

- Motor inertia
- Load
- Drive flexibility



The Torsion Bar Setup v2.0.

The first version of the Torsion Bar (v1.0) setup is equipped with a PWM controlled H-bridge motor amplifier. This corresponds to a velocity-controlled motor. The new Torsion Bar v2.0 contains an ELMO Whiste motor amplifier that can be used in current-control, voltage-control and position-control mode. The Torsion Bar v2.0 setup uses the current control mode, which corresponds to a torque-controlled motor.

Inputs and Outputs

The interface of the ARM 7300 has been limited to the following inputs and outputs:

Digital Inputs

- 2 x LVDS Quadrature Encoder Inputs.

Digital Outputs

- 1 PWM-to-current output, to drive the motor via an ELMO Whistle current amplifier which is mounted below the interface board.
- 2 x LED. Two LED's are present on the board.

Supplier

The Torsion Bar Setup is a product of Controllab Products B.V. Please contact Controllab Products B.V. for more details about this set-up.

Operating System

The Torsion Bar v2.0 setup is delivered with the following operating system:

- RTAI real-time Linux: The setup runs our customized RTAI real-time linux installation including the 20-sim 4C target support software.

7.8 Generic PC or industrial PC/104

The Generic PC or industrial PC is a generic target (typically a general purpose or industrial PC with x86 based CPU like a PC/104 stack) without any specific Input/Output. This target can be used as template for own x86 PC-based targets. Linux supported I/O cards can be added to provide the required inputs and outputs. The following picture shows a PC/104 stack with some mounted boards: a FireWire extension board and a fully configurable FPGA-based digital I/O board.

PC/104 with mounted add on boards.

Inputs and Outputs

No inputs or outputs. External PC I/O board are available from many hardware providers. Prerequisite is that the I/O board can be used under Linux, Comedi or Analogy.

Supplier

Please contact Controllab Products B.V. for suitable hardware providers and possibilities for specific I/O wishes.

Operating Systems

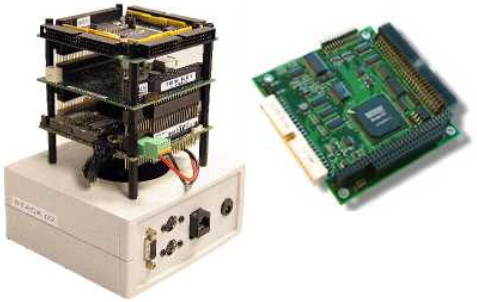
Available options for usage in combination with 20-sim 4C:

- RTAI real-time Linux
- Xenomai real-time Linux

By default, Controllab Products B.V. uses a small embedded Linux distribution with RTAI real-time Linux that includes the 20-sim 4C target support software.
If you want to use 20-sim 4C in combination with your favorite Linux distribution or when you want to use 20-sim 4C in combination with a target running a different operating system than real-time Linux, contact Controllab Products B.V. for the available options.

7.9 PC/104 with Sensoray 526

A generic PC/104 target equipped a Sensoray 526 multi function I/O card to provide Input Output functionality.



PC104 and Sensoray 526 card.

Inputs and Outputs

Analog & Digital Inputs

- Four 24-bit quadrature encoder inputs.
- Eight 16-bit analog inputs.
- 8 x Digital Input. Binary input.

Analog & Digital Outputs

- Four 16-bit analog outputs.
- 8 x Digital Output. Binary output.

Supplier

The Sensoray 526 is a product of Sensoray. More information can be found on the website: http://www.sensoray.com/products/526.htm

Operating Systems

The PC/104 with Sensoray 526 board has been tested with the following operating system:

- RTAI real-time Linux

7.10 TS-7300 - SD-card Installation

Introduction

This page explains how to install a 20-sim 4C image on a SD-card for the TS-7300 board.

Compatibility note: Images for 20-sim 4C 1.1 and 20-sim 4C 2.0/2.1 are not compatible.

20-sim 4C 1.1 only works with a TS-7300 20-sim 4C 1.1 image.

20-sim 4C 2.0 and 2.1 only work with a TS-7300 20-sim 4C 2.0/2.1 image.

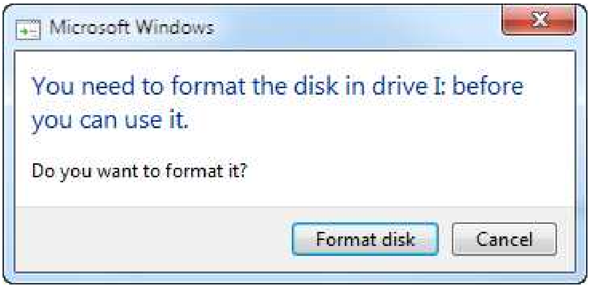
Download

Download 20-sim 4C 1.1 image :http://dl.dropbox.com/u/2230815/TS7300/20-sim-4C-1.1/TS7300-20081107.img.gz

Download 20-sim 4C 2.0 / 20-sim 4C 2.1 image :http://dl.dropbox.com/u/2230815/TS7300/20-sim-4C-2.0/ts7300-r14149.img.gz

Windows Installation

- Extract the image with an archiver (e.g. 7-zip)
- Download the Image Writer for Windows tool from https://launchpad.net/win32-image-writer
- Insert your SD-card in the card-reader. (Write down the drive letter, in this example I:)
- If windows asks to format the SD-card press NO



You need to format the disk in drive I: before you can use it.

Do you want to format it?

- Run the Win32DiskImager.exe as **administrator**.
- Select the image and the correct Drive letter and press the write button

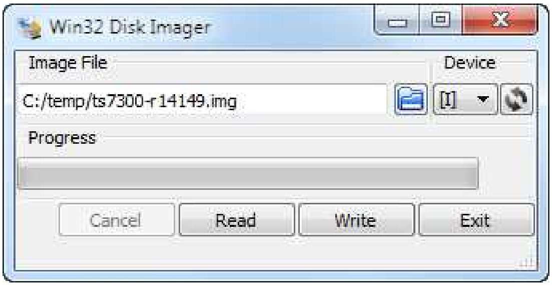


Image File: C:/temp/ts7300-r14149.img

Device: [I]

Linux Installation

- Extract the image with gunzip
- Insert your SD-card in the cardreader (check e.g. with dmesg the device name)
- write the image to the sd-card with dd
  *dd if=ts7300-r14149.img of=/dev/sd**X***
  (where **X** is the location of the SD-Card)

7.11 Upgrading Target Configuration File

Introduction

The target configuration file format in 20-sim 4C 2.1 has been changed. This document describes the changes that need to be made to upgrade an existing 20-sim 4C 2.0 tcf to a 20-sim 4C 2.1 tcf. Although the tcf format is backward compatible tcf files from 2.0 may not work in 2.1. The tcf file defines how and which source files are used, parsed, configured and compiled. 20-sim 4C 2.1 has changed to allow multiple sources (20-sim, Matlab, Scilab) and has multiple targets (Bachmann, TS-ARM, ...) and the process of configuration and compilation has been generalized.

The following sections of the TCF need to be updated.

- GENERAL section
- COMPILERASSISTANT section
- COMPONENTS section

GENERAL section

VERSION

Updated the <VERSION> tag to 2.1.0

```
<VERSION>2.1.0</VERSION>
```

The version tag is used by 20-sim 4C to check whether this tcf is compatible or not.

ICON

This is a new tag that will allow to use a target specific icon in the target overview dialog. This tag may be omitted.

```
<ICON>Bachmann.ico</ICON>
```

FILES_TO_COPY

This tag defines which files are to be copied. This tag is backward compatible and will copy the files, however the TOKENPARSER tag is preferred in 20-sim 2.1.

If the TOKENPARSER section is to be used make this tag empty. (An additional file for a custom target may be added in this tag)

```
<FILES_TO_COPY />
```

FILES_TO_PARSE

This tag defines which files are to be parsed. This tag is backward compatible and will parse the files, however the TOKENPARSER tag is preferred in 20-sim 2.1.

If the TOKENPARSER section is to be used make this tag empty. (An additional file for a custom target may be added in this tag)

```
<FILES_TO_PARSE />
```

FILES_TO_TARGET

For Bachmann targets the content of this tag has been changed.

- There is support for multiple modules at the same time, hence the 'xxsim.m' module must not be hard-coded anymore.
- A model configuration file is necessary. (This is required for logging)

```
<FILES_TO_TARGET>
  <FILE>%FC_MODELBUILDDIR%\%FC_MODELNAME%.m</FILE>
  <FILE>%FC_MODELDIR%\%FC_MODELNAME%.mcf</FILE>
</FILES_TO_TARGET>
```

TOKENS

The TOKENS tag has been changed instead of sequence of TOKEN tags a sequence of TOKENPARSER tags will be used. (Note TOKEN tags may still be given but are ignored)

The TOKENPARSER tag defines a configuration file for the Token parser that will copy and parse the required source files. Note that the TOKENPARSER tag makes the FILES_TO_COPY and FILES_TO_PARSE tags superfluous.

```
<TOKENS>
  <TOKENPARSER>TokenParser.xml</TOKENPARSER>
</TOKENS>
```

INCLUDES

For Bachmann targets, the content of this tag has been changed and now holds a number of required files to be included.

```
<INCLUDES>
  <![CDATA[
  #include <mtypes.h> /* M1 include files */
  #include <msys_e.h>
  #include <mio.h>
  #include <mio_e.h>
  #include <res_e.h>
  #include <svi_e.h>
  #include <log_e.h>
  #include <prof_e.h>
  #include <xxsim_int.h>
  ]]>
</INCLUDES>
```

COMPILERASSISTANT section

The content of this section has been changed for all targets. The toolchains for the various targets is no longer located in the 'contrib' directory but in the 'toolchain' directory, hence the COMMAND needs to be updated.

Replace 'contrib' with 'toolchain' for all the commands.

COMPONENTS section

The components section holds the configuration for each Input/Output module (component). Additional tags have been added in 20-sim 4C 2.1 for better isolation of components, make a components self-containing and removing the need of entries in the GENERAL section.
The following tags have been added and are optional for backward compatibility :

Name	Description
<INCLUDES>	includes
<CONSTRUCT>	Code that is called once to construct this component
<DESTRUCT>	Code that is called once to destruct this component

NOTE: The Bachmann target configuration uses the <TERMINATE>, <CONSTRUCT> and <DESTRUCT> tag for all I/O components. Make sure you update your own configuration accordingly to make sure all I/O components are properly initialized and deinitialized.

8 Operating Systems

8.1 Controllab Discovery daemon

For the automated discovery of targets on the network, most targets run a small program called the Controllab discovery daemon. The Controllab discovery daemon listens on port 1501 (UDP) for broadcast messages send by 20-sim 4C. When it receives a broadcast message from 20-sim 4C, it will reply with a message containing its IP-address and the target name and description as stored on the target.

Supported Operating Systems

The *Controllab Discovery daemon* is currently provided for:

- Linux (for testing purposes; non real-time)
- RTAI real-time Linux
- Xenomai real-time Linux

For usage on other operating systems, contact Controllab Products B.V. for the available options.

Connection details

The *Controllab Discovery daemon* uses port 1501 (TCP) for communication on the TCP/IP network.

8.2 Controllab XMLRPC daemon

To make a target work properly in combination with 20-sim 4C, it should have some kind of an operating system. In order to communicate with the target, each target should run small program on that operating system, called the *Controllab XMLRPC daemon.* The *Controllab XMLRPC daemon* enables the communication between 20-sim 4C and the target.

Supported Operating Systems

The *Controllab XMLRPC daemon* is currently provided for:

- Windows (for testing purposes; non real-time)
- Linux (for testing purposes; non real-time)
- RTAI real-time Linux
- Xenomai real-time Linux

For usage on other operating systems, contact Controllab Products B.V. for the available options.

Connection details

The *Controllab XMLRPC daemon* uses port 1502 (TCP) and 1580 (HTTP) for communication on the TCP/IP network.

8.3 RTAI real-time Linux

RTAI stands for Real-Time Application Interface. It is a real-time extension for the Linux kernel - which allows you to write Linux hosted applications with strict timing constraints.

Version

- RTAI 3.2 on a Linux 2.4.37.x kernel is supported for ARM 7300
- RTAI 3.6, 3.7.x and 3.8.x, 3.9.x are supported on the corresponding Linux 2.6 kernels

Hardware

RTAI supports several architectures:

- x86 (with and without floating point unit (FPU) and timestamp counter (TSC) and on many multi-core CPUs)
- x86-64
- PowerPC
- ARM (StrongARM; ARM7: clps711x-family, Cirrus Logic EP7xxx, CS89712, PXA25x, ARM9: Cirrus Logic EP93xx, Atmel AT91SAM, ARM Cortex)
- MIPS

Supplier

- Like Linux itself the RTAI extension is a community effort. More information on RTAI can be found at: http://www.rtai.org.
- Contact Controllab Products for more information about using 20-sim 4C and RTAI on your own targets.

8.4 Xenomai real-time Linux

Xenomai is a real-time development framework cooperating with the Linux kernel, in order to provide a pervasive, interface-agnostic, hard real-time support to user-space applications, seamlessly integrated into the GNU/Linux environment.

Version

- Support for Xenomai 2.4.10, Xenomai 2.5.x and Xenomai 2.6.x in combination with 20-sim 4C 2.1 is available.

Hardware

Xenomai supports several architectures:

- ARM
- Blackfin
- NIOS II
- PowerPC
- x86

Supplier

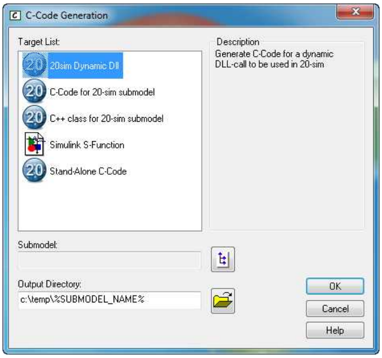
- Like Linux itself the Xenomai framework is a community effort. More information on Xenomai can be found at: http://www.xenomai.org.
- Contact Controllab Products for more information about the usage of 20-sim 4C and Xenomai on your own targets.

9 Troubleshooting

9.1 Errors

I do not see the 20-sim 4C target

If you are generating C-code in 20-sim and see that 20-sim 4C is missing in the targets lists, you have to add the target manually.

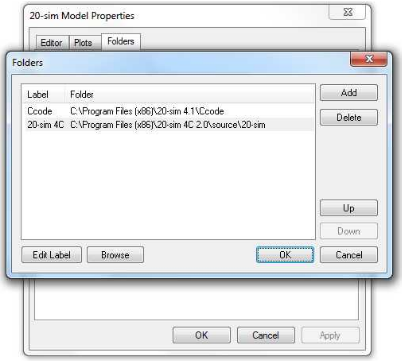


Description: Generate C-Code for a dynamic DLL-call to be used in 20-sim

Output Directory: c:\temp\%SUBMODEL_NAME%

The 20-sim 4C target is missing.

1. In the 20-sim *Editor* from the **Tools** menu select **Options**.
2. Click the **Folders tab** and click the **C-code Folders button**.
3. Click the **Add button** and browse to the 20-sim 4C folder:



Label	Folder
Ccode	C:\Program Files (x86)\20-sim 4.1\Ccode
20-sim 4C	C:\Program Files (x86)\20-sim 4C 2.0\source\20-sim

Add the 20-sim 4C code generation templates to 20-sim.

The default location is:

```
C:\Program Files\20-sim 4C 2.1\source\20-sim
```

or for 64-bit Windows versions:

```
C:\Program Files (x86)\20-sim 4C 2.1\source\20-sim
```

I cannot find the log file

If you cannot find the log file for use in 20-sim please check:

- Did you select variables for logging in 20-sim 4C?
- Did you select Run with Logging in 20-sim 4C?
- Click the Configure Logging tab to see what filename and location you have chosen in 20-sim 4C.

Configure Button is red

If the Configure button stays red, no connection could be made to the target.

1. Make sure the target is powered on. If a fatal crash has occurred, switch the power off for and after a few seconds switch it on again. Then try to search for an IP-address again.
2. Make sure you have a cross cable or network cable connected to your PC and your target.

20-sim 4C does not find an IP-address

Check the Network section for help.

I see multiple IP-Addresses

Check the Network section for help.

Compile Button is red

If the Compile button stays red, compilation was not finished successfully. Check if the right target was chosen.

9.2 Network

20-sim 4C does not find an IP-address

All network related problems result in the same issue, 20-sim 4C does not find an IP-address. Please check the following conditions :

1. First of all, make sure that a network is connected between your PC and your target. A cross cable if connected directly between your target and your PC, a network cable if connected via a network switch or hub.
2. Check if the Target is connected and is powered on.
3. Check if the led's at both network connections are on (is the network cable inserted well?).
4. Check with your network administrator of system administrator that the network is not blocking traffic between your PC and your target. Turn-off your firewall temporarily to check if this is the problem.
5. If multiple networks are connected to your PC, disable all networks that are not connected to your target.

6. If no DHCP server is used, the target will fallback on the automatic private address of 169.254.254.254. Make sure the network interface on the host pc has a network IP-address in the same range.

Cross Cable

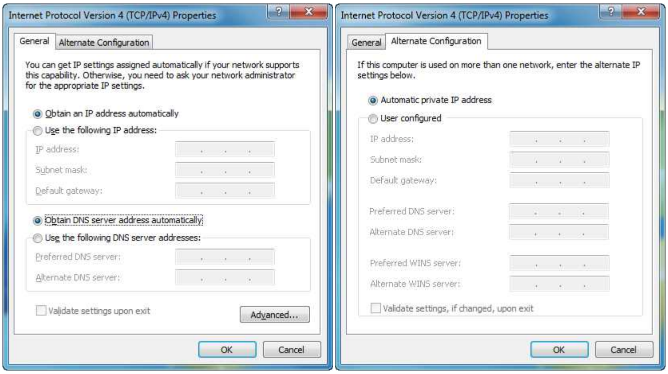
If all conditions are met but 20-sim 4C does not find an IP-address you can try to use a cross-cable to connect the target with your PC.

1. First of all, make sure that a cross cable is connected between your PC and your target. A cross cable if connected directly between your target and your PC, a network cable if connected via a network switch or hub.

2. Check if the target is connected and is powered on.

3. Open the **TCP/IP Properties** of your connection. Go to Windows Help and search for "configure TCP/IP settings" to find out how you can open the TCP/IP Properties.

4. Make sure the following settings are chosen.



You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

● Obtain an IP address automatically
○ Use the following IP address:

● Obtain DNS server address automatically
○ Use the following DNS server addresses:

□ Validate settings upon exit

If this computer is used on more than one network, enter the alternate IP settings below.

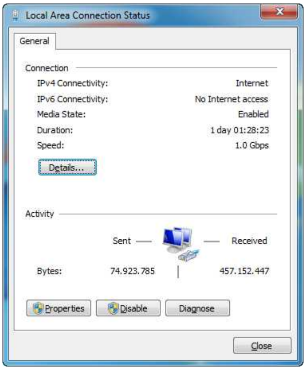
● Automatic private IP address
○ User configured

□ Validate settings, if changed, upon exit

The proper settings for the TCP/IP properties of you connection.

5. Open the ***Local Area Connection Status*** window of your connection. Go to Windows Help and search for "local area connection" to find out how you can open the *Local Area Connection Status* window.

The status of your network adapter should look like the figure below.



Local Area Connection Status

General

Connection

IPv4 Connectivity:	Internet
IPv6 Connectivity:	No Internet access
Media State:	Enabled
Duration:	1 day 01:28:23
Speed:	1.0 Gbps

Activity

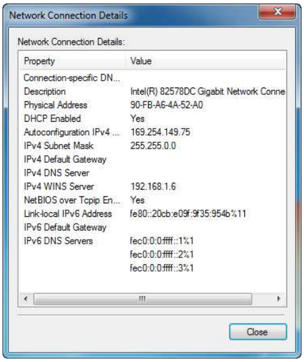
	Sent	Received
Bytes:	74.923.785	457.152.447

The status of your network.

Do not worry about the warning. Windows will try to obtain a automatic private address. This may take some time (1 minute in some cases) before the connection is established.

6. Click on the **Details** button.

If the private address was obtained successfully the IP address should be in the range 169.254.x.x.



Network Connection Details

Network Connection Details:

Property	Value
Connection-specific DN...	
Description	Intel(R) 82578DC Gigabit Network Conne
Physical Address	90-FB-A6-4A-52-A0
DHCP Enabled	Yes
Autoconfiguration IPv4 ...	169.254.149.75
IPv4 Subnet Mask	255.255.0.0
IPv4 Default Gateway	
IPv4 DNS Server	
IPv4 WINS Server	192.168.1.6
NetBIOS over Tcpip En...	Yes
Link-local IPv6 Address	fe80::20cb:e09f:9f35:954b%11
IPv6 Default Gateway	
IPv6 DNS Servers	fec0:0:0:ffff::1%1 fec0:0:0:ffff::2%1 fec0:0:0:ffff::3%1

Check your IP address.

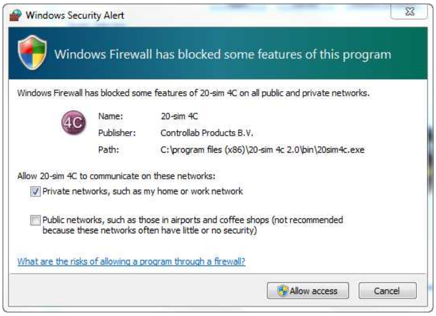
7. **Close** all network windows and return to 20-sim 4C.

If after one minute a connection cannot be made, try the following steps:

- Turn-off your firewall temporarily to check if this is the problem.
- Close all other network connections (do not forget your wireless network) to check if this is the problem.
- Enter the IP address 169.254.254.254 manually in 20-sim 4C.

My Firewall gives a Warning

If you click the **Refresh** button of the **Configure** Target tab, 20-sim 4C will scan the network for connected targets. An active firewall will respond by giving a warning message:



Windows Security Alert

Windows Firewall has blocked some features of this program

Windows Firewall has blocked some features of 20-sim 4C on all public and private networks.

Name: 20-sim 4C
Publisher: Controllab Products B.V.
Path: C:\program files (x86)\20-sim 4c 2.0\bin\20sim4c.exe

Allow 20-sim 4C to communicate on these networks:

☑ Private networks, such as my home or work network

□ Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing a program through a firewall?

The Windows firewall message when the Refresh button is clicked.

You should allow 20-sim 4C access to the network to find the connected targets. If you do not allow access, the correct IP-address can not be shown.

1. Click the **Allow Acces** button (or a similar button if another firewall message is shown) if the name of the program is 20-sim 4C and the publisher is Controllab Products B.V.

2. Please contact your **network administrator** if the **name** of the program is not 20-sim 4C or the **publisher** is not Controllab Products B.V.

3. Please contact your **network administrator** if you do not have **permission to change the firewall settings**.

What to do with multiple IP-Addresses

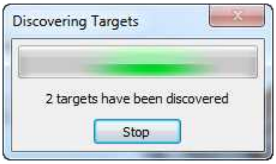
20-sim 4C automatically searches for targets which are connected to a network. If more than one IP-address is listed, it means that several targets are connected to the network. To find out the address of a target, if more than one address is listed do the following:

1. Switch the **power** on of your target.

2. In 20-sim 4C click on the **Configure** button.

3. In the *Configure Target* tab, Click the **Discover** button**.**

20-sim 4C will now start a search to find the IP-addresses of connected targets.

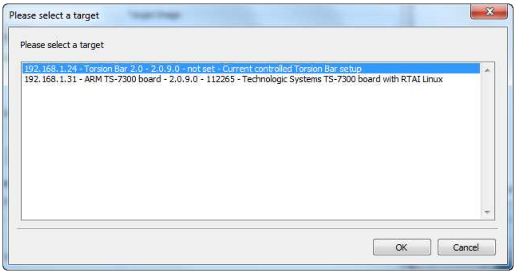


Discovering Targets

2 targets have been discovered

Finding connected targets.

After a few seconds the search is ready and a list of targets is shown.



Please select a target

192.168.1.24 - Torsion Bar 2.0 - 2.0.9.0 - not set - Current controlled Torsion Bar setup

192.168.1.31 - ARM TS-7300 board - 2.0.9.0 - 112265 - Technologic Systems TS-7300 board with RTAI Linux

Select the target that you would like to use.

4. **Select** your target and click **OK**.

In most networks, IP-addresses are automatically assigned. In most networks, once an address is assigned, it will stay the same. If you are not sure about the assignment of addresses in your network, please contact your network administrator.

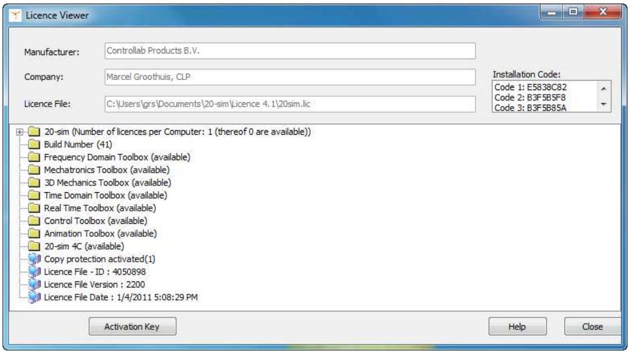
9.3 Software

General

1. Check if the Operating System of your PC is Windows XP, Windows Vista or Windows 7, Windows 8.
2. Check if your PC is connected to a network or that a cross cable is plugged in.

20-sim

1. **Open 20-sim** and check if the program is properly working, e.g. load a example model and try to simulate it.
2. In **20-sim** from the **Help** menu choose **Register / Update License....** This opens the *Registration/Update License* dialog.
3. In the *Registration/Update License* dialog select the **License Viewer** button. This opens the *License Viewer*.



Manufacturer: Controllab Products B.V.

Company: Marcel Groothuis, CLP

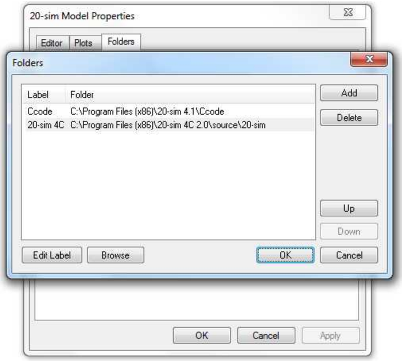
Licence File: C:\Users\grs\Documents\20-sim\Licence 4.1\20sim.lic

Installation Code:
Code 1: E5838C82
Code 2: B3F5B5F8
Code 3: B3F5B85A

20-sim (Number of licences per Computer: 1 (thereof 0 are available))
Build Number (41)
Frequency Domain Toolbox (available)
Mechatronics Toolbox (available)
3D Mechanics Toolbox (available)
Time Domain Toolbox (available)
Real Time Toolbox (available)
Control Toolbox (available)
Animation Toolbox (available)
20-sim 4C (available)
Copy protection activated(1)
Licence File - ID : 4050898
Licence File Version : 2200
Licence File Date : 1/4/2011 5:08:29 PM

The 20-sim License Viewer.

4. Check if all toolboxes are available (especially the **Real Time Toolbox**). When not all toolboxes are available, contact you local distributor or contact Controllab Products to obtain a 20-sim Professional license.
5. In 20-sim from the **Tools** menu click the **Options** command. In the window that pops up, click the **C-code Folders** button on the **Folders** tab. Check if the right folders are shown.
7. Follow the exercise "Running a Test model". If 20-sim 4C opens after the code generation, 20-sim is correctly installed.
8. **Note:** depending on the location of 20-sim 4C, the path may be different. Enter you own path here if you installed 20-sim 4C on a non-standard location. when you are generating C-code from 20-sim, the code will be automatically transferred to 20-sim 4C.



Label	Folder
Ccode	C:\Program Files (x86)\20-sim 4.1\Ccode
20-sim 4C	C:\Program Files (x86)\20-sim 4C 2.0\source\20-sim

Add the 20-sim 4C code generation templates to 20-sim.

9. Follow the exercise "Running a Test model". If 20-sim 4C opens after the code generation, 20-sim is correctly installed.

Index

20-sim 4C is a prototyping environment that allows you to export C-code to hardware like PC-104 systems and ARM processor boards.

This allows you to perform various tasks:
Measurement: You can run a code block that reads sensor signals.
Analysis: You can log data and import it in 20-sim for analysis.
Control: You can create a machine controller in 20-sim and export it to hardware.

The name 4C stands for Configure, Connect, Compile and Command and these are the tasks that you have to perform to get code running on a target.

ISBN 978-90-79499-14-4
9 789079 499144

20-sim controllab 20-sim 4C

Controllab Products B.V. Hengelosestraat 500, 7521 AN Enschede, The Netherlands
T +31(0)85 773 18 72, E info@controllab.nl, **www.20sim.com**