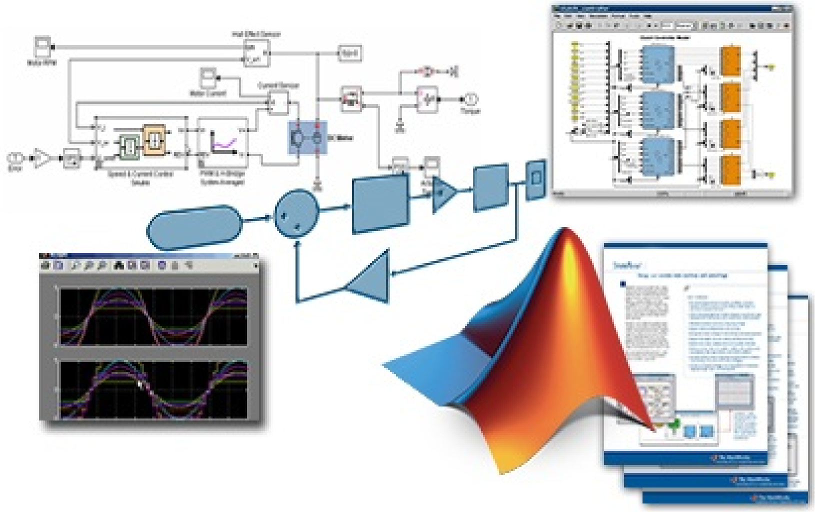
Reader: xPC Getting Started

version : 2.4
date : 13 April 2015
by : W.Nijs and M.Driessen
tested on : R2013b



Index

1. Introduction

xPC-Target is a development environment for designing, constructing and testing prototypes, using a *Real-Time operating system (xPC-OS)* running on *standard PC hardware (xPC-target)*. This development environment is a member of the *Matlab/Simulink Product Family*. It uses, in addition to a host PC on which the development environment is running, a target PC to control real-time applications..

The environment runs on a standard computer serving as a host PC with *Matlab*, *Simulink* and *Stateflow* (optional), to design and simulate (not in real-time) models using *Simulink blocks* and *Stateflow charts*.

The xPC-Target toolbox allows you to add I/O blocks to your model and to convert them into executable code using *Real-Time Workshop*, *Real-Time Workshop Embedded Coder* (optional), *Stateflow Coder* (optional) and a C/C++ compiler (*Microsoft Visual Studio*). The executable code is downloaded to the target PC where the *xPC Target Real-Time kernel* is running at that time. After downloading, the application/program can be run and tested in real-time.

At Fontys Mechatronics Engineering Venlo the standard xPC-targets are equipped with an ASUS P3B-F motherboard, an Intel PRO/100+ network adapter, a CD-ROM drive for booting and a colour monitor as Target Scope. The BIOS settings is set to force booting from the CD-ROM drive and to disable error messages when the mouse and/or keyboard are not connected.

2. Configuration

As mentioned above, in most cases the configuration consists of one *Host* and one *Target* PC. The Host PC is used as *development environment* (Matlab/Simulink) and the target PC is used for *running the developed application program*.

The Host and Target PC are connected through TCP/IP (required for downloading the application programme and retrieving test data from the target PC). There are two option for the I/O to the application:

a. The target is equipped with PCI-cards (PCI-6024E and/or PCI-QUAD04, see 10.2, 10.3 and 10.4).

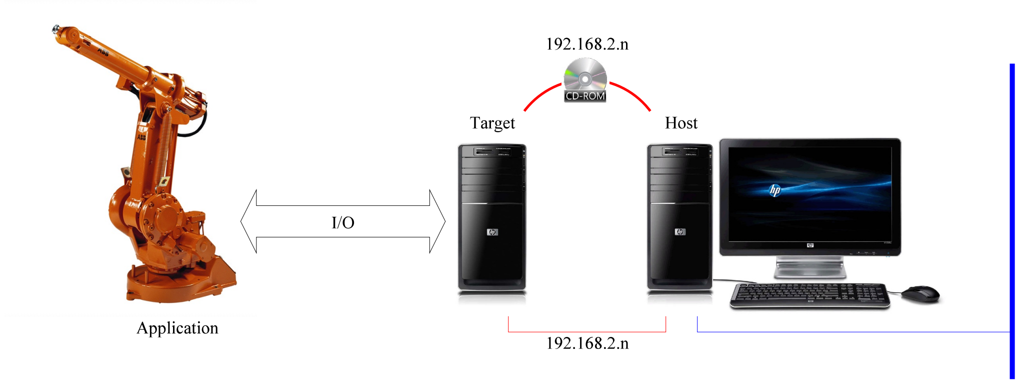


Fig.2a Host-Target configuration with PCI-cards for I/O to the application.

b. The target makes use of an iMIA (intelligent Motion Interface Adapter, see 10.5).

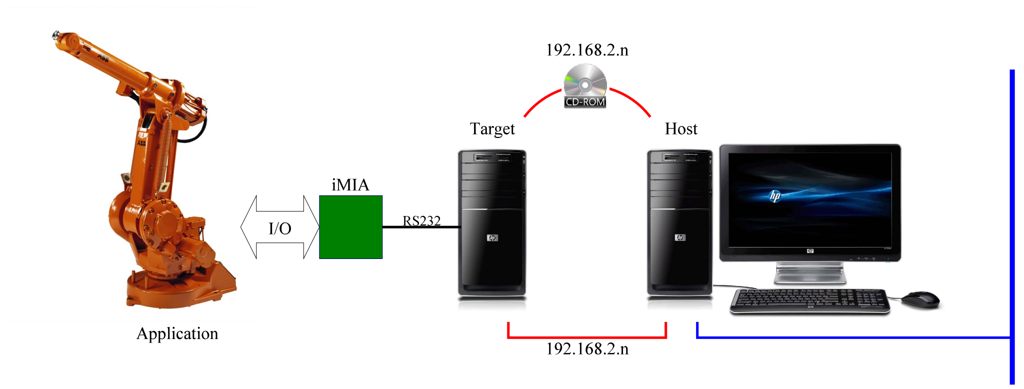


Fig.2b Host-Target configuration with the iMIA for I/O to the application.

The Target PC is booted with a CD-ROM containing the Real-Time xPC OS (Operating System). This boot CD-ROM is (in most cases) the same for each Target PC (see 7.2. Boot configuration).

3. Preparing the Host

Note: This chapter should only be followed when a Fontys Host-PC is used. The current Laptop Policy aims that you use your own laptop. Instructions how to install, configure and test the Microsoft, Mathworks and Java software can be found in the reader: *Installatie en test van Matlab/Simulink met xPC.*

3.1 The Local Netwerk

The reason for a local network is, that the Target PC's are not accepted by the routers of the Fontys network. So we have to install and configure a second NIC

Note: For the following operations you need admin rights.

Configuratiescherm -> Netwerkstatus en -taken weergeven -> Adapterinstellingen wijzigen -> LAN-verbinding 2

Eigenschappen -> Internet Protocal versie 4 (TCP/IPv4) -> Eigenschappen

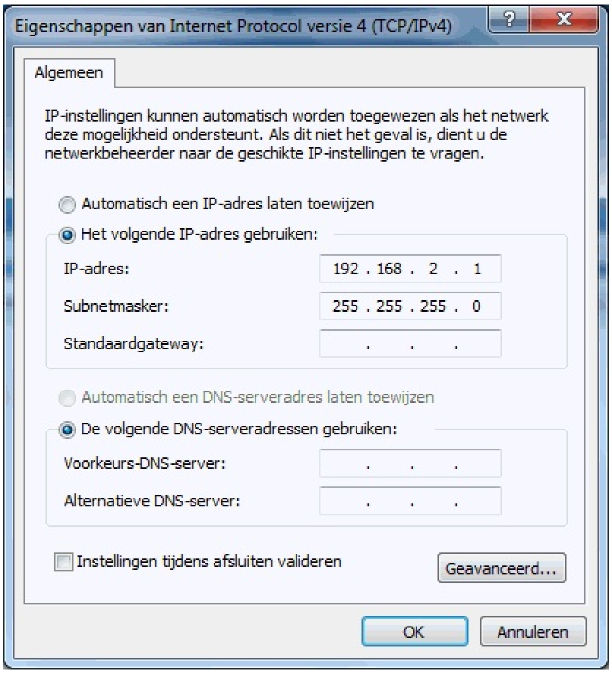


Fig.3a Local network configuration.

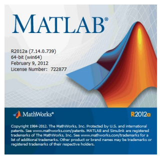
Select: Het volgende IP-adres gebruiken:

IP-adres : 192.168.2.1
Subnetmasker : 255.255.255.0

Leave everything else blank.

Note: The choice of this IP address is directly related to the settings of xPC Target Explorer (and the boot CD-ROM, see 7. XPC Target Explorer).

3.2 Compiler Path



To build executable code for your model and/or application for running it on the xPC Target, Matlab/Simulink wants to know where to find the compiler.

From the Matlab prompt run the command:

```
>> xpcsetCC('setup')

Select your compiler for xPC Target.

[1] Microsoft Visual C++ Compilers 2010 Professional in C:\Program
Files (x86)\Microsoft Visual Studio 10.0

[0] None


Compiler:1

Verify your selection:

Compiler: Microsoft Visual C++ Compilers 2010 Professional
Location: C:\Program Files (x86)\Microsoft Visual Studio 10.0

Are these correct [y]/n?y

Done...
```

This setting should be saved as part of your user profile, so you have to do this only once.

4. Preparing the xPC Target

4.1 Bios

Set the BIOS so that the Target boots without restrictions from the CD-ROM drive.

4.2 Booting xPC-OS

You need a boot medium. Normal a CD-ROM is used, but a floppy is also possible. This boot medium is (in most cases) the same for each Target PC and can be created with xPC Target Explorer (see 7.2. Boot configuration).

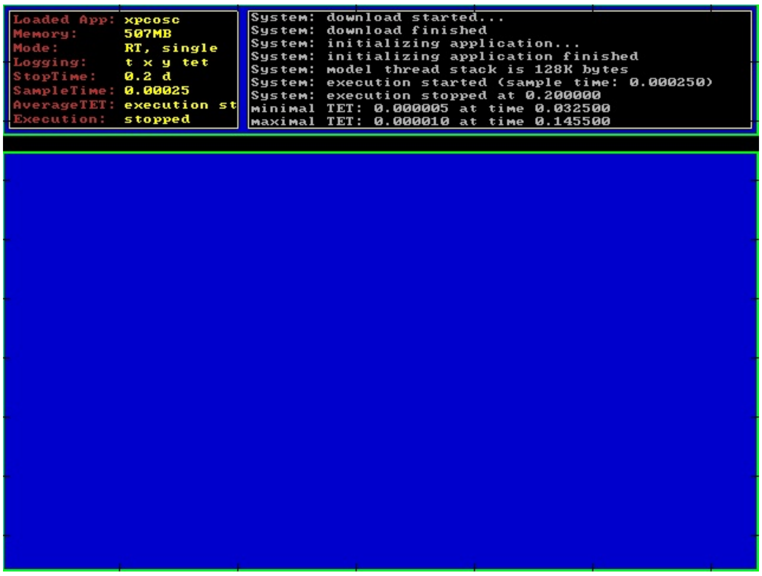


Fig.4a The Target monitor after a (re)boot.

The graphical user interface (GUI):

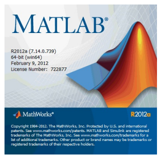
```
>> xpctargetspy
```

allows you to upload displayed data from the target PC. The behavior of this function depends on the value for the environment property TargetScope.

- If TargetScope is enabled, a single graphics screen is uploaded. The screen is not continually updated because of a higher data volume when a target graphics card is in VGA mode. To update the host screen with another target screen, move the pointer into the Spy window and left-click.

- If TargetScope is disabled, text output is continuously transferred every second to the host and displayed in the window.

4.3 Configuration Test



With the *xpctest* command a series of xPC Target tests wil be started to check the functioning xPC Target. The tasks are:

- Initiate en test the communication between the Host and Target computers.
- Reboot the Target to reset the target environment.
- Build the target application on the Host.
- Download the target application to the Target.
- Check communication between the Host and the Target using commands.
- Execute a target application.
- Compare the results of a simulation and the target application run

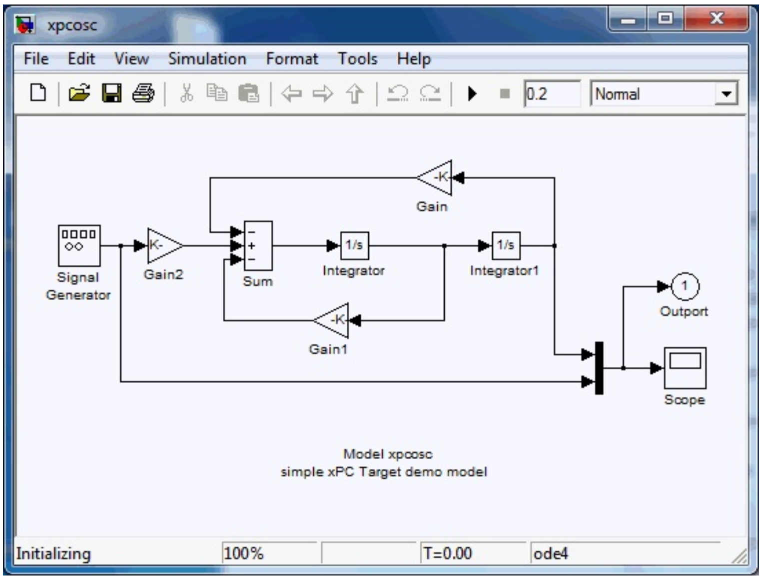


Fig.4b The xpctest application model.

From the Matlab prompt run the command:

```
>> xpctest

### xPC Target v5.2 Test Suite
### Host-Target interface is: TCP/IP (Ethernet)
### Test 1, Ping target PC 'TargetPC1' using system ping: ... OK
### Test 2, Ping target PC 'TargetPC1' using xpctargetping: ... OK
### Test 3, Software reboot the target PC 'TargetPC1': ..... OK
### Test 4, Build and download an xPC Target application using model
xpcosc to target PC 'TargetPC1': ... OK
### Test 5, Check host-target command communications with 'TargetPC1':
... OK
### Test 6, Download a pre-built xPC Target application to target PC
'TargetPC1': ... OK
### Test 7, Execute the xPC Target application for 0.2s: ... OK
### Test 8, Upload logged data and compare with simulation results: ...
OK
### Test Suite successfully finished
```

When errors occur, please refer to Chapter 7. XPC-Target Explorer

5. Simulink Models

Simulink models are represented graphically as block diagrams. A wide array of blocks are available to the user in provided libraries for representing various phenomena and models in a range of formats. One of the primary advantages of employing Simulink (and simulation in general) for the analysis of dynamic systems is that it allows us to quickly analyze the response of complicated systems that may be prohibitively difficult to analyze analytically. Simulink is able to numerically approximate the solutions to mathematical models that we are unable to, or don't wish to, solve "by hand."

5.1 Simulating a Model on the Host

> **Note:** If you are working on a host computer of Fontys, always set the location of any Simulink models that you create or use to a project directory on your Z: drive. There are insufficient privileges for another location on the host PC.

As example a simple model (derived from The MathWorks Quick Start 2-6) is available on the ELO or can be entered.

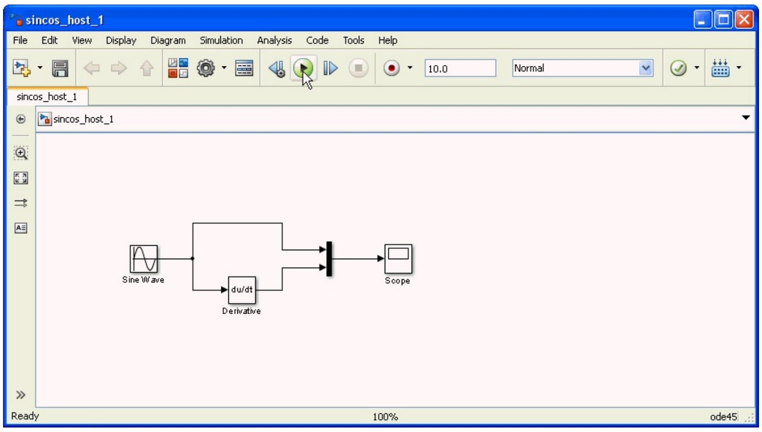


Fig.5a The Simulink model for simulating on the Host.

File -> New -> Model

The used symbols can be found in with the Library Browser:

Sine Wave	: Sources
Derivative	: Continuous
Mux	: Common Used Blocks
Scope	: Common Used Blocks

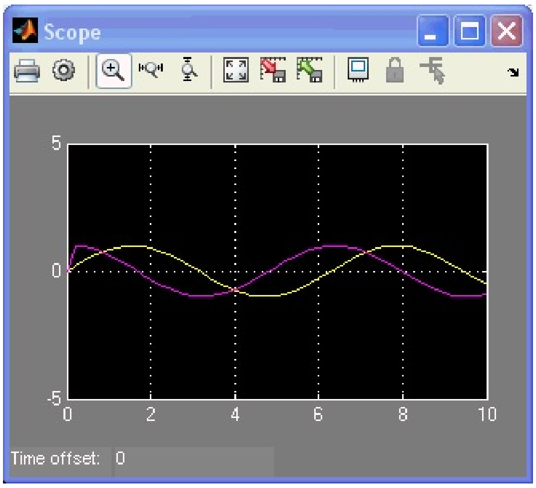


Fig.5b The simulation results.

5.2 Running a Model on the Target

The same example as in 5.1 is used, only with some small changes.

Note: The *Target Scope* (added in the block diagram) makes it possible to view the results also on the target monitor (see: Block library -> xPC -> Misc).

Note: Set the model choice to External (default Normal).

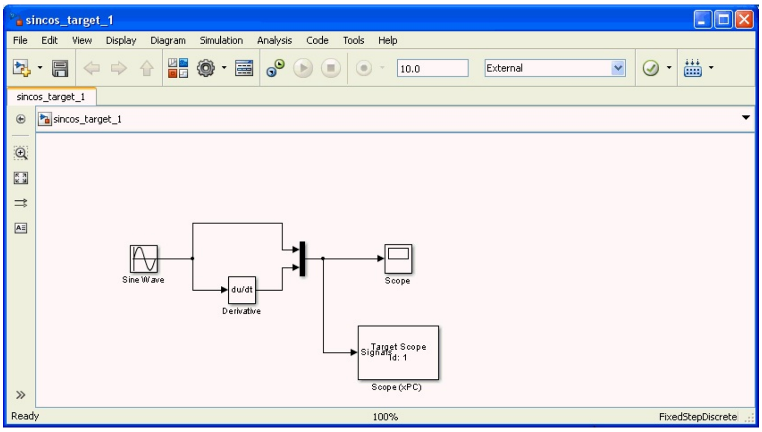


Fig.5c The Simulink model for running on the Target.

Before you can make a real-time run of your model/application on the Target, you have to:

- Set the real-time configure parameters for your model/application (see: 6.)
- Set the Target properties (see: 7.)
- Build and download the executable code of your model/application.
- Make connection to the Target and start (and stop) the real-time run.

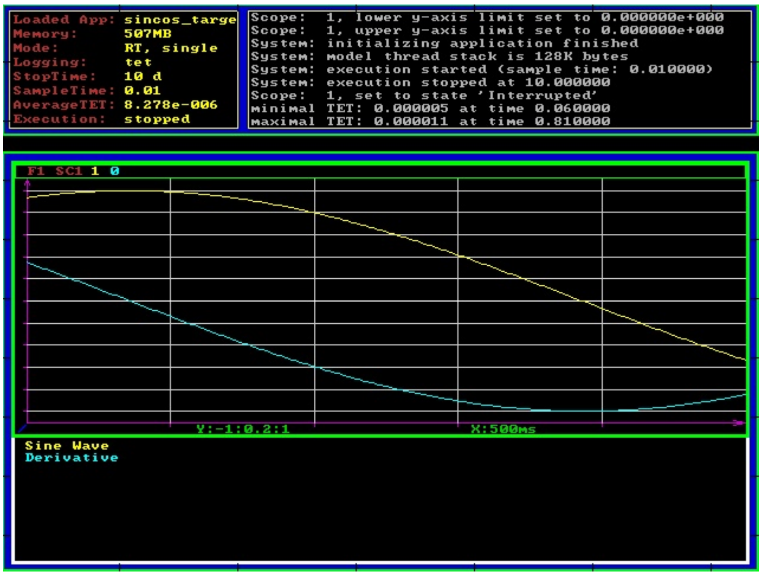


Fig.5d The Target monitor with the real-time results.

6. Configuration Parameters

After designing or importing the Simulink model, always check the configuration parameters:

Simulation → Model Configuration Parameters (Crt-E)

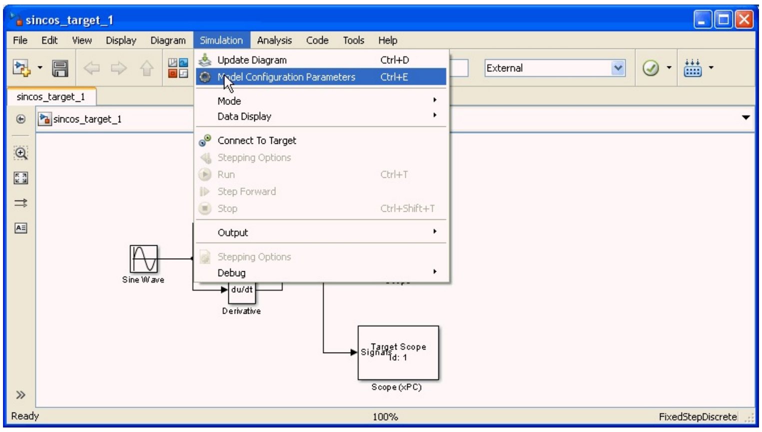


Fig.6a To the *Model Configuration Parameters*.

6.1. Solver

Select: Solver: ode3 (Bogacki-Shampine)

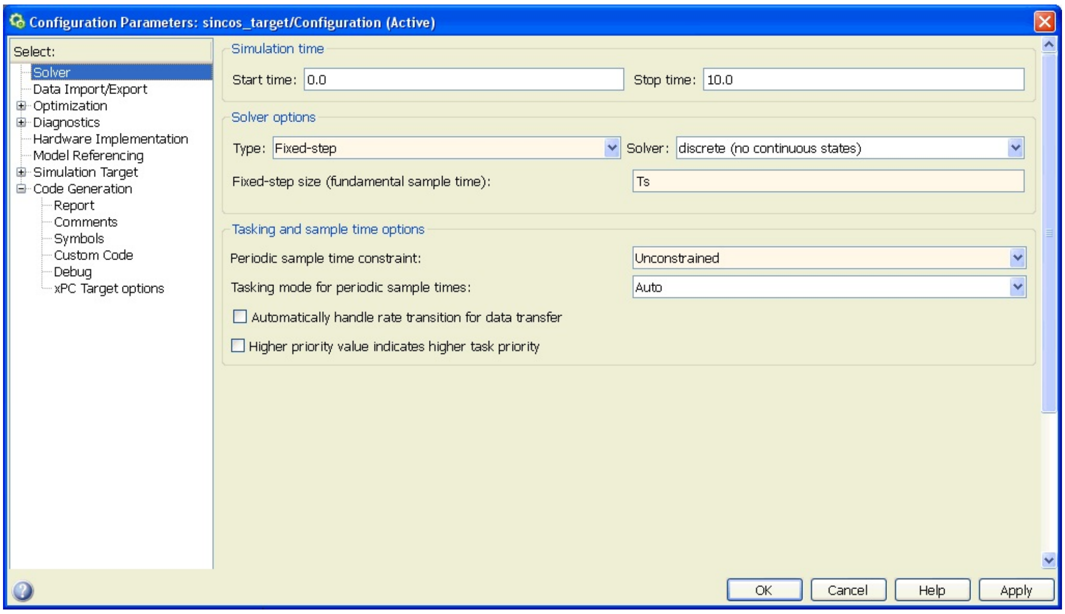


Fig.6b Solver settings.

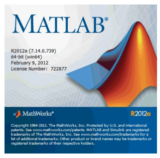
Simulation time

The simulation start and stop time (and thus duration) are set here

Start time: 0.0
Stop time: 10.0

Solver options

The parameter Fixed-step size (fundamental sample time) can be replaced by one variable, e.g. *Ts*. The advantage to this is that when multiple processes are running, they all get the same sample time.



Note: Don't forget that this variable *Ts* also needs to be set in MATLAB itself, e.g.:

```
>> Ts = 0.01
```

Note: The selection of Start time, Stop time and Fixed-step size therefore determine the total number of samples (see also 9 Data readback in MATLAB).

6.2 Code Generation

In Configuration Parameters select: Code generation:

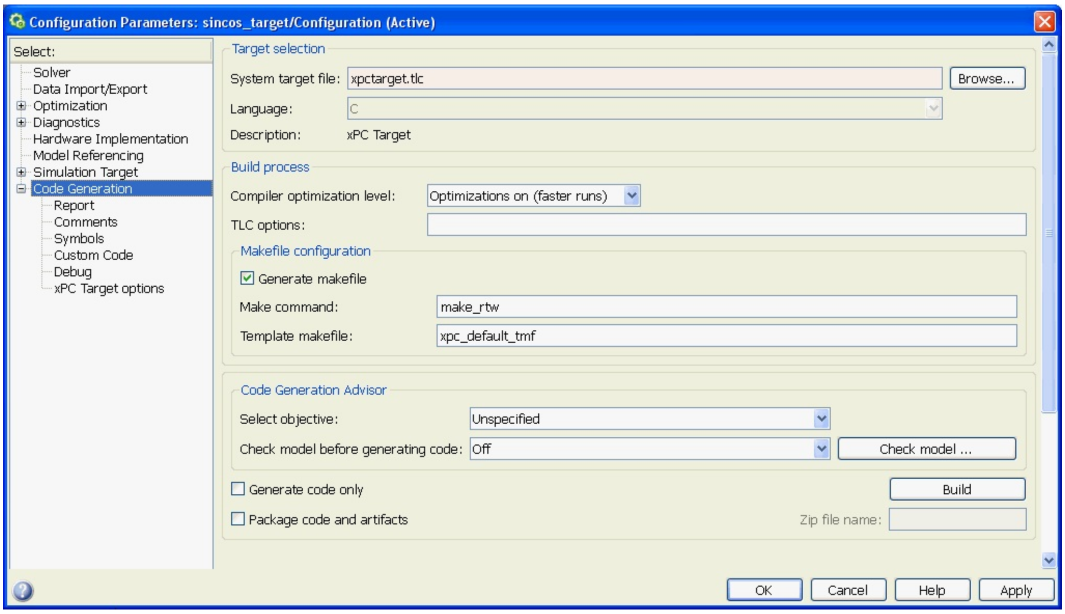


Fig.6c Code Generation settings.

System target file

If this is not set for your xPC Target, then select via:

→ Browse.... → select: xpctarget.tlc

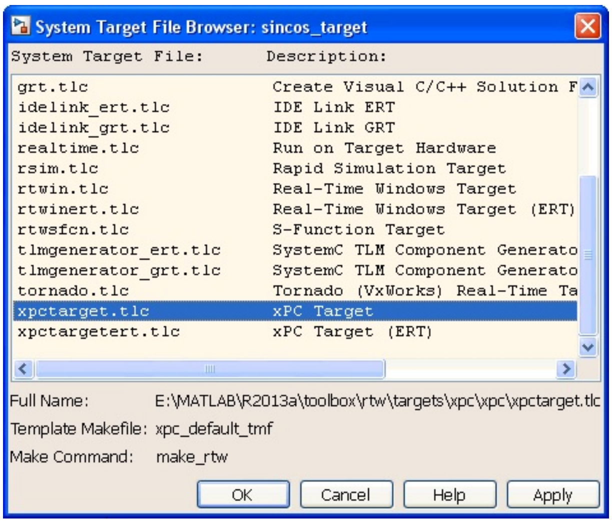
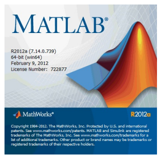


Fig.6d Target selection.

7. xPC-Target Explorer

The setup for the Target (TargetPC1) can be adjusted with the xPC Target Explorer started up with the following MATLAB command:




```
>> xpcexplr
```

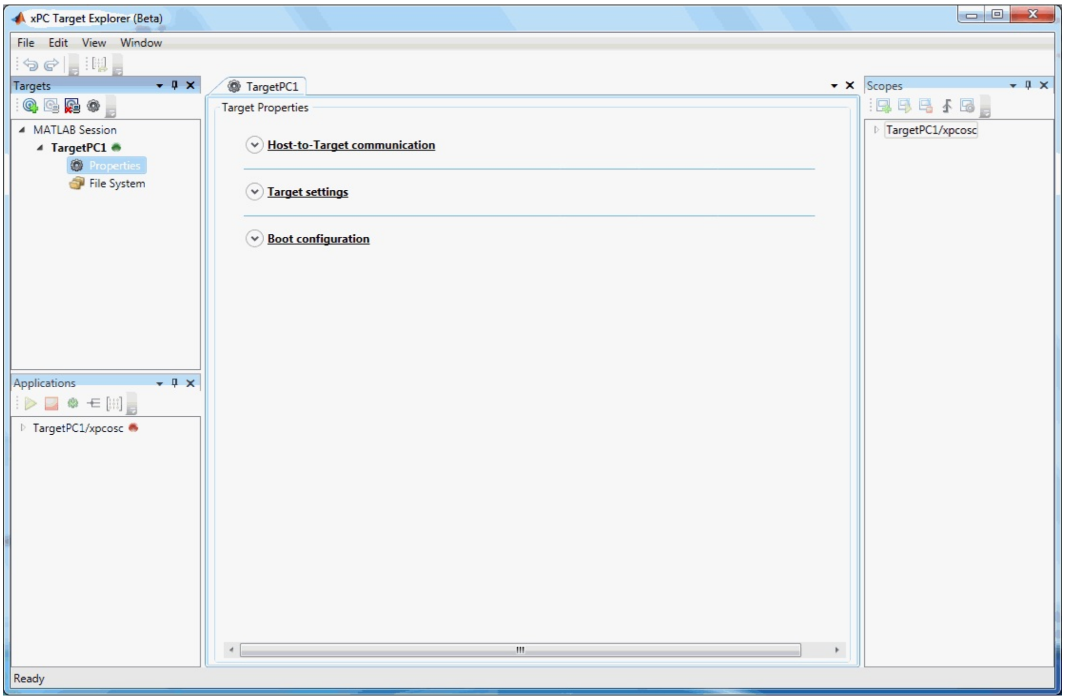


Fig.7a TargetPC1 Properties

7.1. Communication

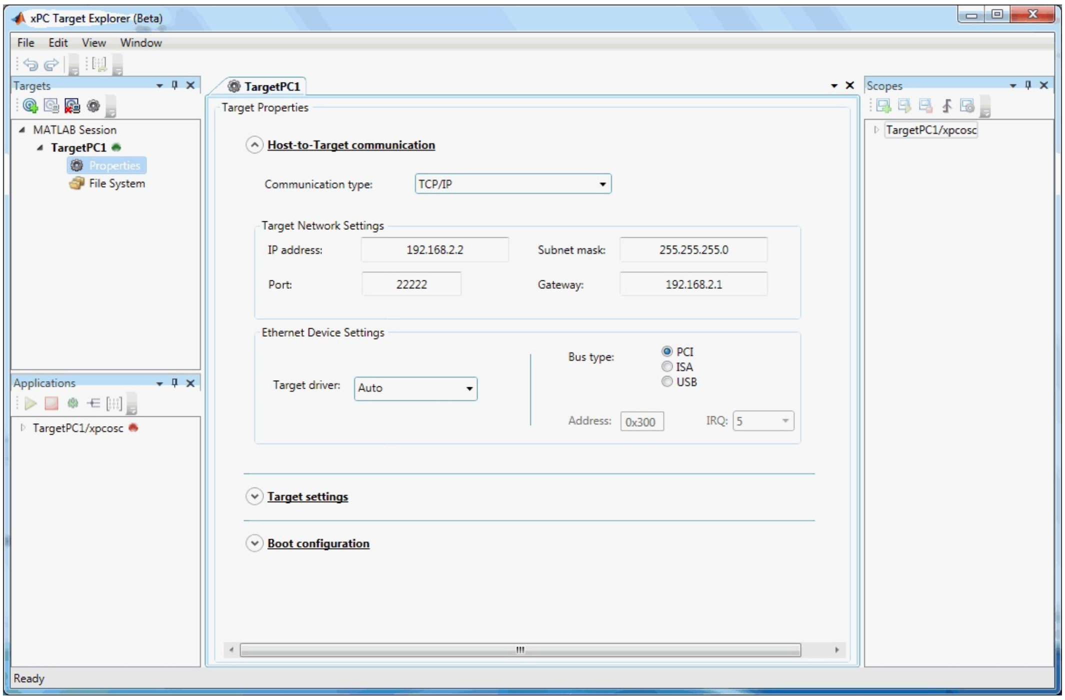


Fig.7b Host-to-Target communicatie

Adjust the following settings, if necessary:

Communication type : TCP/IP

IP address : 192.168.2.2

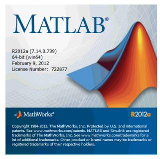
Note: This IP address is (in most cases) the same for each target (see also 5.3. Creating a boot disk).

Subnet mask : 255.255.255.0
Port : 22222
Gateway : 192.168.2.1

Bus type : PCI

The adjusted parameters may be subsequently be changed and are sticky, meaning that once changed they are saved as default for the *TargetPC1* for all Simulink models.

Communication with the target can be tested in MATLAB window with the command:



>> !ping 192.168.2.2 (as escape to Windows)

or:

>> xpctargetping

7.2. Boot Configuration

For most target PC's used in the labs a boot CD-ROM is created. If required for special cases (different IP address, other xPC OS-version, target without CD-ROM drive), a different boot medium can be created under:

Boot configuration:

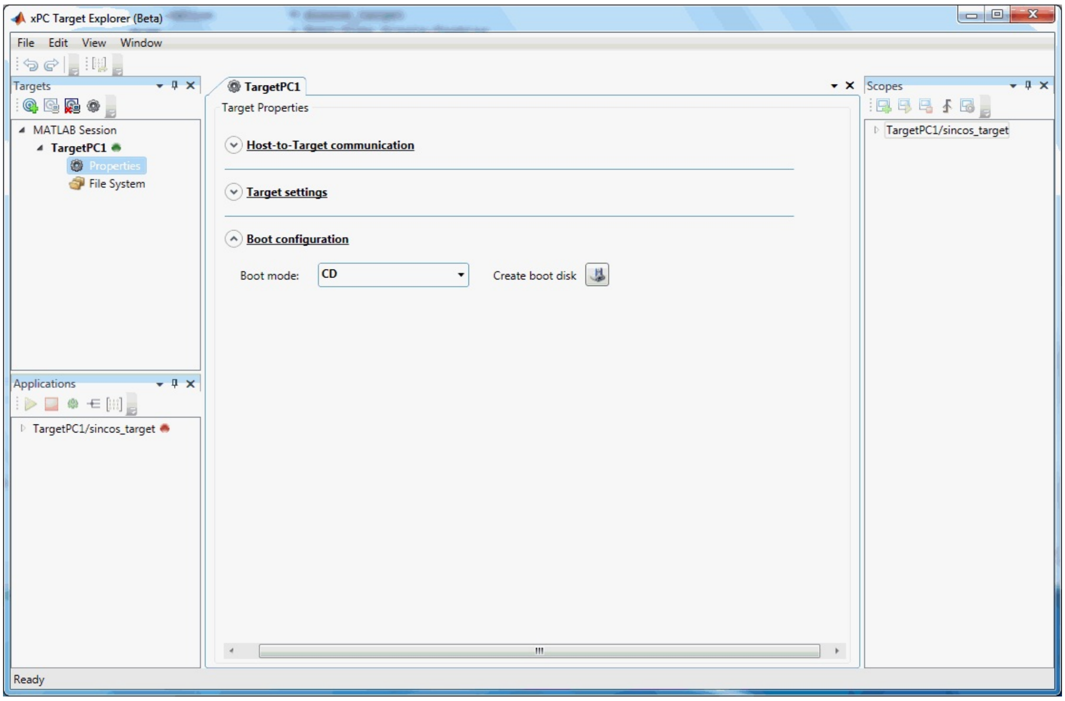


Fig.7c Creating a boot CD-ROM

8. Building and Running

If you want to run a Simulink model on a xPC-target, the model have to *translated to C-code* first, then *compiled and linked* and at least *downloaded as machine code* to the target. This process is known as *builden*:

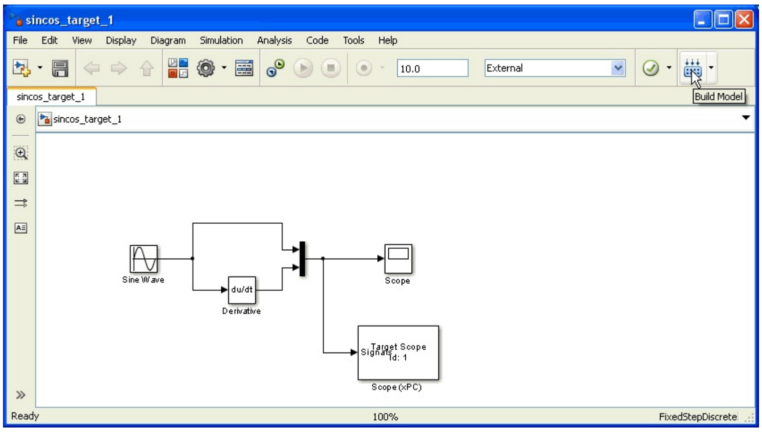


Fig.8a The build process.

When the application model is stored as machine code on the target, *Connect To Target* with the target:

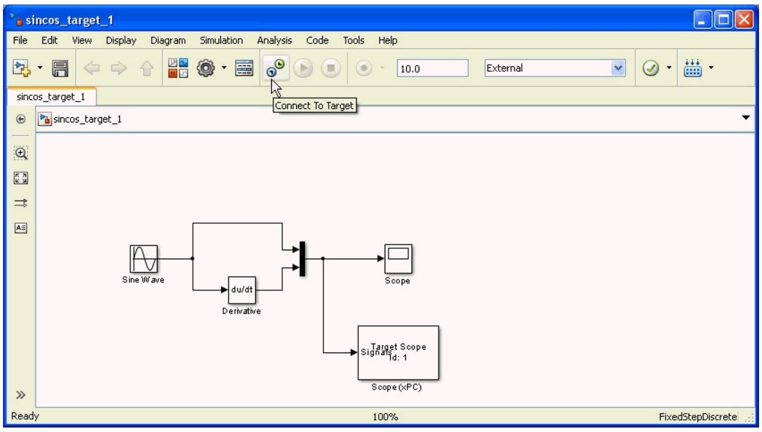


Fig.8b Make the connection to the target.

The last step is to *start* the application (running):

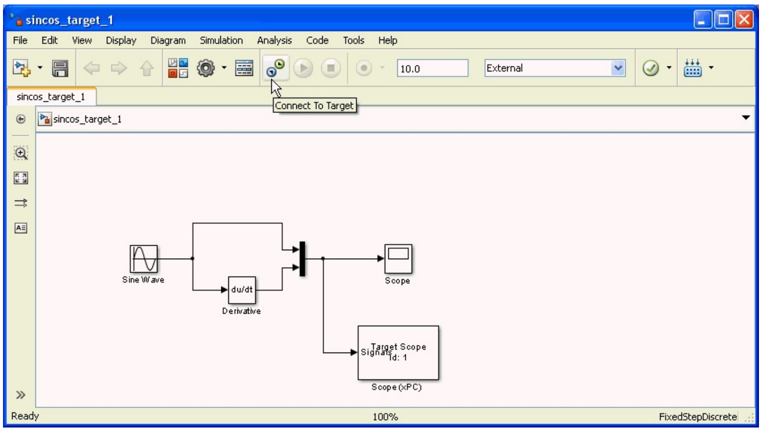


Fig.8c Start the application.

9. Capturing Real-Time Data for MATLAB

When measured results should be compared with simulated results, we have to create the possibility, by modifying the model (9.1) and configuration parameters (9.2), to send the captured results to the Host for plotting/processing (9.3).

9.1. Customizing the Model

First, you have to place a Out symbol, preceded by a multiplexer symbol, to make it possible to capture multiple signals. By using the Clock symbol gives the possibility also write the sample moments to the Out array.

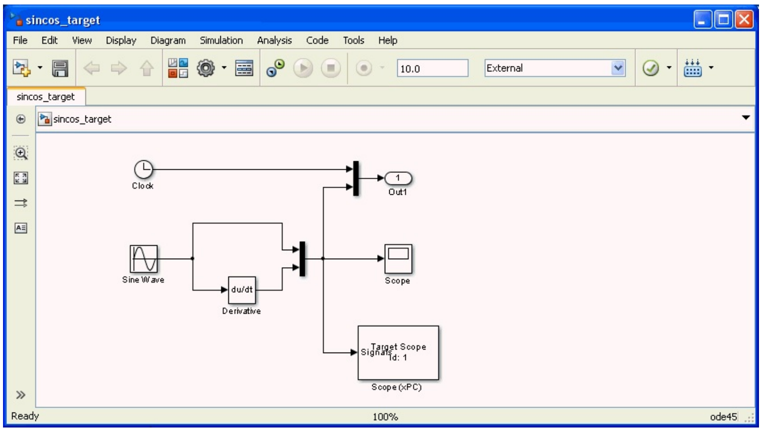


Fig.9a Capturing data for Matlab.

9.2 Customizing the Configuration Parameters

Second, check at the Configuratie Parameters → Data Import/Export if the Save to workspace *Output* option is selected.

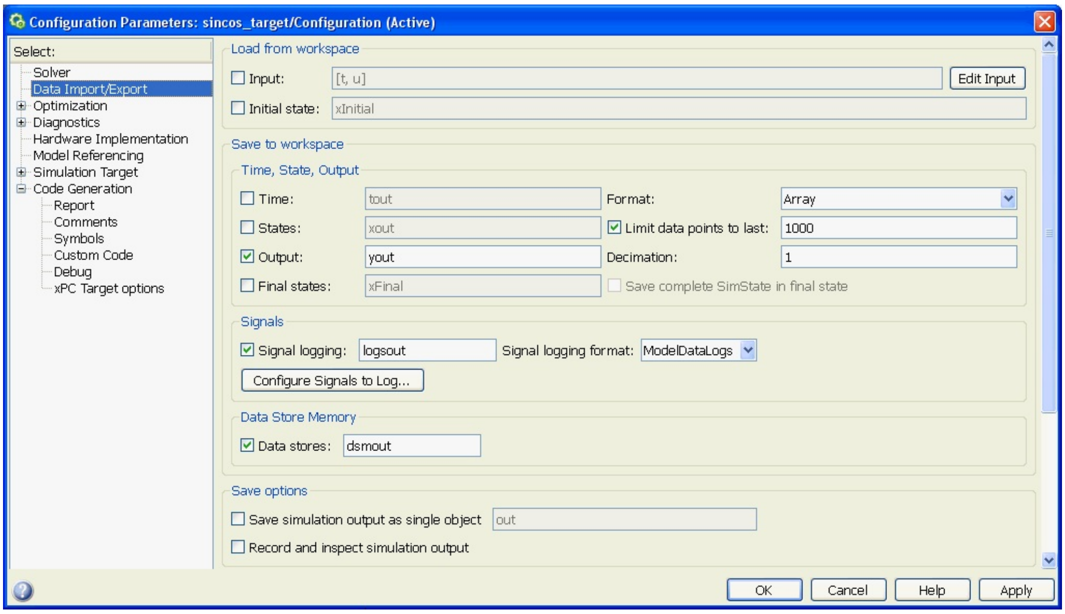


Fig.9b Settings for Data Export (Save to workspace).

9.3 Processing Real-Time Data within MATLAB

Within the MATLAB-window we can import the real-time data from the xPC target with the command:

```
>> result_1 = tg.OutputLog
```

Note: The real-time data is overwritten after every run, so choice an other variable name (result_2, result_3 etc.) for your data.

```
result =

         0         0         0
    0.0100    0.0100    1.0000
    0.0200    0.0200    0.9999
    0.0300    0.0300    0.9997
    0.0400    0.0400    0.9994
    0.0500    0.0500    0.9990
    0.0600    0.0600    0.9985
    0.0700    0.0699    0.9979
    0.0800    0.0799    0.9972
    0.0900    0.0899    0.9964
    0.1000    0.0998    0.9955
```

And e.g. plotted with the command:

```
>> plot (result_1(:,1),[result_1(:,2),result_1(:,3)])
```

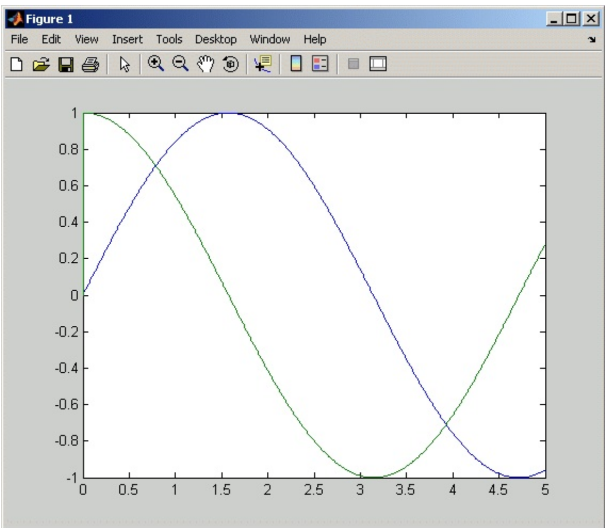


Fig.9c The plotted result in Matlab.

In MATLAB, you can also take over the vectors with:

```
>> t = result_1(:,1)
>> x = result_1(:,2)
>> y = result_1(:,3)
```

and plot them with:

```
>> plot (t,[x y])
```

Note: Always use the same sample (Ts), Start and Stop time whenever the simulation and measurements results should be compared to one another. Only then the vectors are of equal lenght and can be processed in one graph without a problem.

To save a variable from the Workspace to disk (to compare data in a next session) use:

```
>> save result_1
```

To load a variable from disk to the Workspace (to compare data from a previous session) use:

```
>> load result_1
```

10. Interface Cards, PCI Busses and Slots

If you are using xPC Target in a real-time testing environment, you require a target computer with I/O modules/cards.

10.1. What's where

Available I/O cards at Fontys Mechatronics Engineering Venlo are:

- PCI-6024E with 16(/8) ADC's, 2 DAC's, 8 binairy in/outputs, 1 counter (PWM)
- PCI-6221M with ......
- PCI-QUAD04 with 4 quadrature incremental shaftencoder interfaces/counters

Source Block Parameters (see also Fig.10e and Fig.10h).

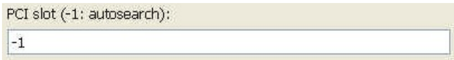


Fig.10a Autosearch the PCI slot.

If you use more as one of the same type I/O cards in your target, Simulink is not in the case to do this automaticly. To find out which I/O cards on which PCI busses and in which slots they are installed, the following command can be used:

```
>> getxpcpci
```

...... result for CFT-robot ......

To force the PCI slot parameter at the Source Block Parameters you have to change -1 (autosearch) in the by the card used bus and slot information e.a. [7,9] for buss 7 and slot 9.

Fig.10b Force the PCI slot by [7,9]

10.2. PCI-6024E (National Instruments)

Fig.10c The PCI-6024E I/O card.

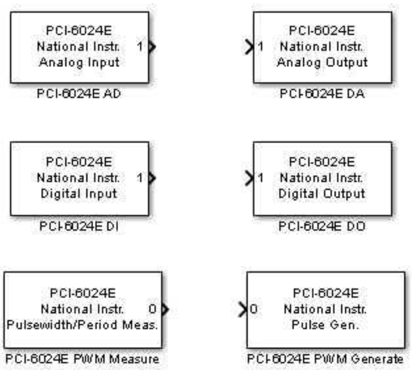


Fig.10d The 6 PCI-6024E symbols.

Fig.10e The PCI-6024E Breakout Board.

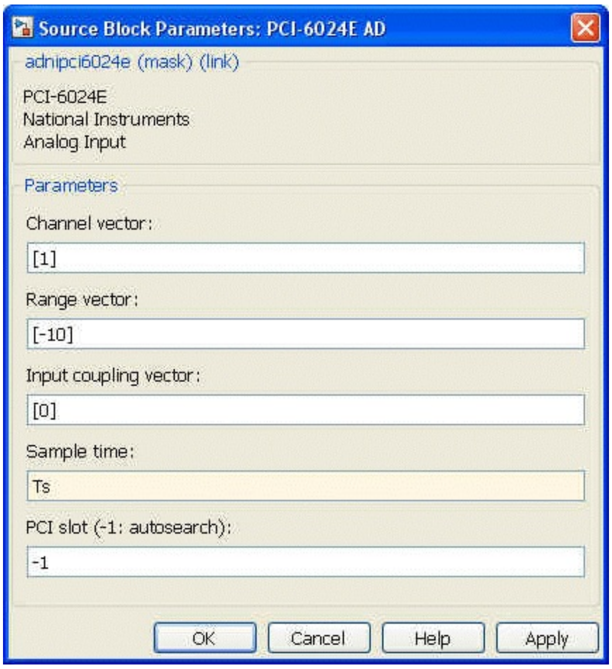


Fig.10f The Analog Input settings.

Fig.10g The Direction (52), PWM (2) and Digital Ground (18, 36) pins.

10.3. PCI-6221M (National Instruments)

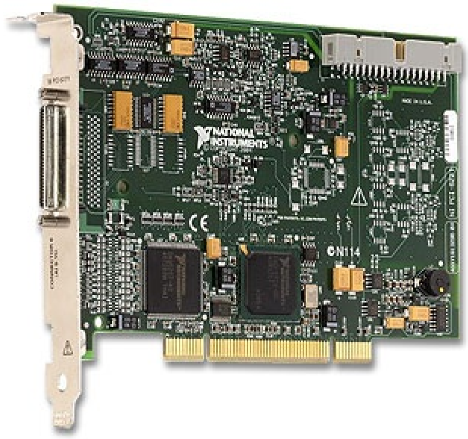


Fig.10h The PCI-6221M I/O card.

10.4. PCI-QUAD04 (Measurement Computing)

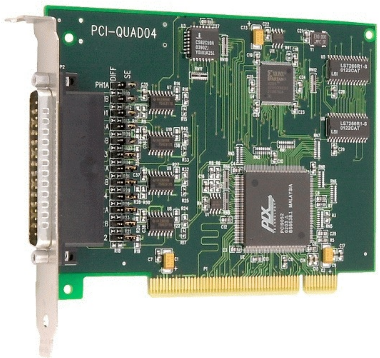


Fig.10i The PCI-QUAD04 I/O card.

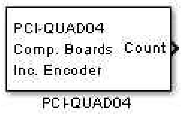


Fig.10j The PCI-QUAD04 symbol.

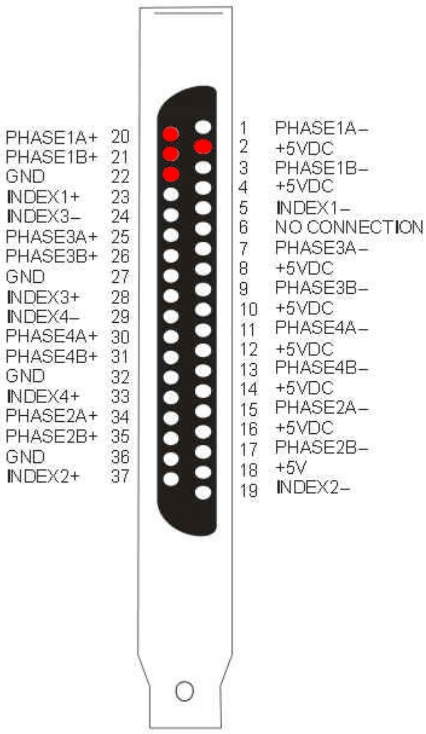


Fig.10k The PCI-QUAD04 SE Channel 0 connection.

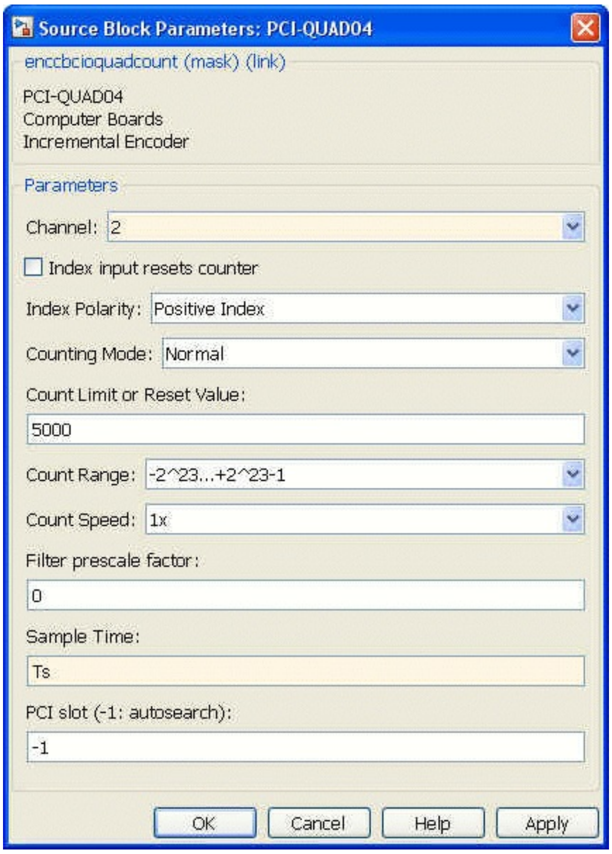


Fig.10l The Incremental Encoder settings.

HEDS5540/PCI-QUAD04 connection

1	GND	zwart	22
2	IND	wit	N.C.
3	CHA	groen	20
4	+5V	rood	2
5	CHB	geel/blauw	21

11. Intelligent Motion Interface Adaptor (Fontys/ALF4all)

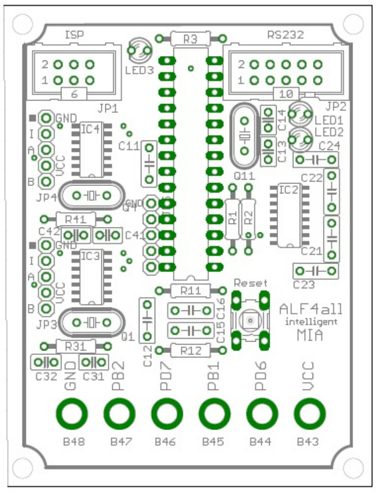


Fig.11a The iMIA I/O card.