ETAS

DRIVING EMBEDDED EXCELLENCE

# ETAS INTECRIO-RLINK V5.0

## Getting Started

www.etas.com

## Copyright



INTECRIO-RLINK V5.0 – Getting Started R03 EN – 05.2022

# Contents

# 1 Safety and Privacy Information

In this chapter, you can find information about the intended use, the addressed target group, and information about safety and privacy related topics.

Please adhere to the ETAS Safety Advice (**Help > Safety Advice**) and to the safety information given in the user documentation.

## 1.1 Demands on the Technical State of the Product

The following special requirements are made to ensure safe operation:

- Take all information on environmental conditions into consideration before setup and operation (see the documentation of your computer, hardware, etc.).

## 1.2 Target Group

This manual is intended for trained personnel specializing in the area of function and software development for embedded electronic systems.

A good working knowledge of MATLAB® and Simulink® and INCA/INCA-EIP is required.

In addition, INTECRIO-RLINK users should be familiar with the operating systems Microsoft Windows® 8, Windows® 8.1, or Windows® 10. All users should be capable of executing menu functions, activating buttons etc. Users should also be acquainted with the Windows file storage system, particularly the relations between files and directories. Users must know about and have mastered the basic functions of the Windows File and Program Manager and Windows Explorer. Users should also be familiar with "Drag & Drop".

All users who are not familiar with the basic techniques of Microsoft should familiarize themselves with them before using INTECRIO-RLINK. Details of working with Windows are contained in the relevant manuals by Microsoft Corporation.

Knowledge of a programming language, preferably ANSI-C, can be helpful to advanced users.

## 1.3 Intended Use

With INTECRIO-RLINK, Simulink® models can be easily and rapidly tested in the vehicle. INTECRIO-RLINK enables users to carry out all steps required to generate executable prototypes directly within Simulink. The ETAS prototyping hardware can be configured directly within the Simulink environment.

The rapid prototyping blockset supports the complete configuration of the various ETAS prototyping targets, including the definition of bypass interfaces, sensor signals, and actuator signals.

With the virtual prototyping blockset of INTECRIO-RLINK, Simulink models can be analyzed without the need for complex prototyping hardware.

ETAS GmbH cannot be made liable for damage which is caused by incorrect use and not adhering to the safety information.

## 1.4 Classification of Safety Messages

Safety messages warn of dangers that can lead to personal injury or damage to property:

**DANGER**

**DANGER** indicates a hazardous situation that, if not avoided, will result in death or serious injury.

**WARNING**

**WARNING** indicates a hazardous situation that, if not avoided, could result in death or serious injury.

**CAUTION**

**CAUTION** indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.

***NOTICE***

**NOTICE** indicates a situation that, if not avoided, could result in damage to property.

## 1.5 Safety Information

Please adhere to the ETAS Safety Advice and to the following safety information to avoid injury to yourself and others as well as damage to property.

**WARNING**

**Wrongly initialized NVRAM variables can lead to unpredictable behavior of a vehicle or a test bench.**

**This behavior can cause harm or property damage.**

INTECRIO-RLINK systems that use the NVRAM possibilities of the experimental targets expect a *user-defined* initialization that checks whether all NV variables are valid for the current project, both individually and in combination with other NV variables. If this is not the case, all NV variables have to be initialized with their (reasonable) default values.

Due to the NVRAM saving concept, this is *absolutely necessary* when projects are used in environments where any harm to people and equipment can happen when unsuitable initialization values are used (e.g. in-vehicle-use or at test benches).

Further safety advice for this ETAS product is available in the following formats:

- In electronic form on the installation medium. See `Documentation\General\ETAS Safety Advice.pdf` for details.

- The "ETAS Safety Advice" window that opens when you start the program, or when you select **Help > Safety Advice**.

## 1.6 Privacy Notice

Your privacy is important to ETAS so we have created the following Privacy Statement that informs you which data are processed in INTECRIO-RLINK, which data categories INTECRIO-RLINK uses, and which technical measure you have to take to ensure the users' privacy. Additionally, we provide further instructions where this product stores and where you can delete personal data.

### 1.6.1 Data Processing

Note that personal data respectively data categories are processed when using this product. The purchaser of this product is responsible for the legal conformity of processing the data in accordance with Article 4 No. 7 of the General Data Protection Regulation (GDPR). As the manufacturer, ETAS GmbH is not liable for any mishandling of this data.

### 1.6.2 Data and Data Categories

Please note that this product creates files containing file names and file paths, e.g. for purposes of error analysis, referencing source libraries, or for communicating with third-party programs.

The same file names and file paths may contain personal data, if they refer to the current user's personal directory or subdirectories (e.g., `C:\Users\<UserId>\Documents\...`).

Furthermore, using ETAS Rapid Prototyping solutions in test vehicles connected to real sensors, buses or ECUs, the ETAS tools may get access to personal data of the driver.

This data can also be stored using dataloggers as provided by INCA-EIP or the ETAS Experiment Environment.

When using the ETAS License Manager in combination with user-based licenses, particularly the following personal data respectively data categories can be recorded for the purposes of license management:

- Communication data: IP address
- User data: UserID, WindowsUserID

### 1.6.3 Technical and Organizational Measures

This product does not itself encrypt the personal data respectively data categories that it records. Ensure that the data recorded are secured by means of suitable technical or organizational measures in your IT system.

Personal data in log files can be deleted by tools in the operating system.

# 2 About INTECRIO-RLINK

With the ETAS *INTECRIO-RLINK* prototyping blockset, Simulink® models can be easily tested onboard a vehicle. With the aid of the powerful, real-time capable ETAS prototyping targets, the behavior of Simulink models can be thoroughly validated under real-world conditions.

For in-vehicle testing, INTECRIO-RLINK supports modular ES800 prototyping hardware, compact ES900 prototyping hardware, and non-real-time prototyping on the Windows PC.

This manual supports the user when getting to know INTECRIO-RLINK to ensure fast results. It provides a step-by-step introduction to the system with all information easy to look up.

## 2.1 System Information

INTECRIO-RLINK supports the following versions of MATLAB® and Simulink®:

- R2016a – R2021b

INTECRIO-RLINK supports the following MATLAB® and Simulink® components:

- Simulink®
- MATLAB® Coder™
- Simulink® Coder™
- Embedded Coder®
- Stateflow®
- Fixed-Point Designer™ (combines Simulink® Fixed Point™ and Fixed-Point Toolbox™)

INTECRIO-RLINK supports experiments with the following tools:

- INCA / INCA-EIP V7.0 or higher
- ETAS Experiment Environment V3.7 or higher

INTECRIO-RLINK can be installed and used independent of other ETAS software (INTECRIO, ASCET-RP, EHOOKS, etc.).

## 2.2 Documentation Structure

The INTECRIO-RLINK documentation consists of the following parts:

- this Getting Started manual
- an online help that describes the Simulink blocks provided by INTECRIO-RLINK (cf. section 2.3)
- an online help that describes how to use the hardware configurator (cf. section 2.4)

When INTECRIO-RLINK is installed (cf. section 3.2.1), the INTECRIO-RLINK Getting Started guide, as well as the INTECRIO-RLINK release notes and the safety advice, can be accessed from the file system at `<installation>\Manuals`.

The INTECRIO-RLINK Getting Started manual consists of the following chapters:

- **"Safety and Privacy Information"**

  In this chapter, you can find information about the intended use, the addressed target group, and information about safety and privacy related topics.

- **"About INTECRIO-RLINK" (this chapter)**

  This chapter gives an initial overview of the possible field of application of INTECRIO-RLINK.

- **"Installation"**

  This chapter is intended for all users who install, maintain and uninstall INTECRIO-RLINK. It provides important information on the delivery scope, hardware and software requirements. The sequence of both the installation and uninstallation of INTECRIO-RLINK is described.

- **"INTECRIO-RLINK Quick Guide"**

  This chapter is a quick introduction to the program concept of INTECRIO-RLINK. The working examples, displayed in the form of flow-charts, provide you with an overview of the program functionality and operating mode.

- **"Bypass Concept"**

  This chapter explains the concept of hook-based and service-based bypass.

- **"Glossary"**

  Abbreviations and specialist terms which occur in this manual are described in the glossary. The terms are listed in alphabetical orders.

- **"Troubleshooting General Problems"**

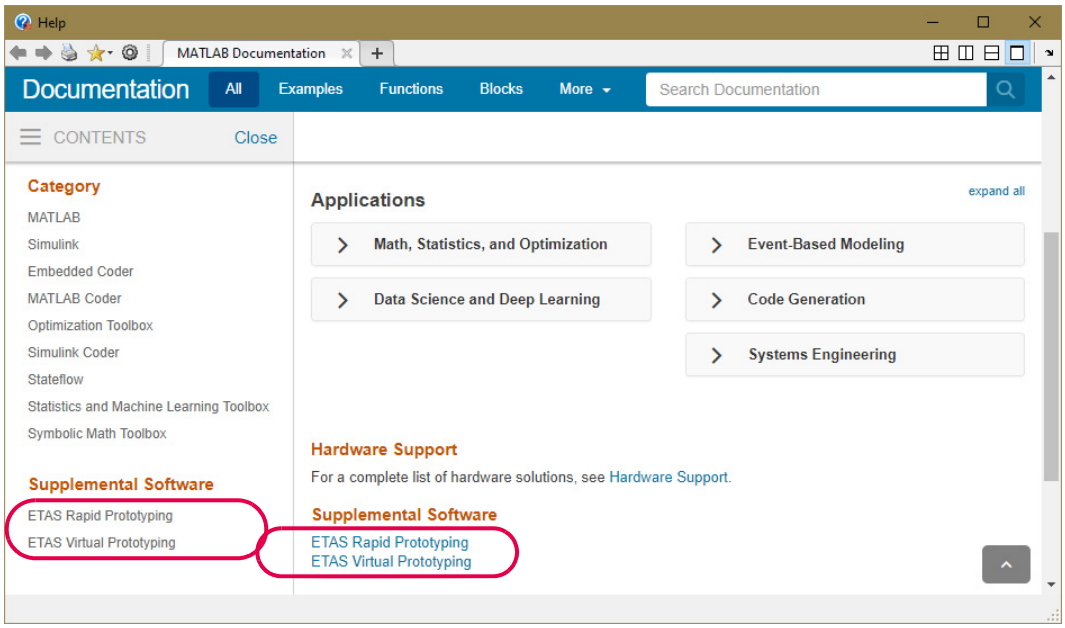
  This troubleshooting chapter gives some information of what you can do when problems arise that are not specific to an individual software or hardware product.

- **"Contact Information"**

  This chapter contains ETAS contact addresses.

## 2.3 INTECRIO-RLINK: Online Help

The INTECRIO-RLINK online help is integrated in the MATLAB and Simulink online help.




## 2.4 INTECRIO-RLINK: Hardware Configurator Online Help

Use the **Help > Help** menu option or the button to invoke the general help function. Press the <F1> function key to call context-sensitive help.


The tabs of the help window provide you with the following options:

- The "Contents" tab allows you to browse the help topics by categories.
- The "Index" tab lists all index entries. Browse the entire list, or enter a search term to limit the scope of listing.
- The "Search" tab allows you to search for individual words or terms included in a help topic. Type a search string and let the help function list the entries it has found related to this term.
- The "Favorites" tab allows you to bookmark topics.

# 3 Installation

This chapter is for all users who install, maintain, or uninstall INTECRIO-RLINK. This chapter provides information on the delivery scope, hardware and software requirements for the installation, and the preparation required for installation. The chapter also describes the procedures used to install and uninstall INTECRIO-RLINK.



## 3.1 Preparations

Check the delivery package to make sure it is complete and make sure your system corresponds to the system requirements. Make sure that you have the necessary user privileges.

### 3.1.1 Delivery Scope

The delivery scope of INTECRIO-RLINK includes

- installation disk
  - INTECRIO-RLINK program files
    i.e. program files for rapid prototyping and virtual prototyping
  - ETAS Virtual OS Execution Platform program files
    required for virtual prototyping
  - INTECRIO-RLINK Getting Started manual in PDF format[1]
  - INTECRIO-RLINK release notes, product overview etc. in PDF format[1]
  - ETAS hardware documentation in PDF format[1]
  - Safety hints in PDF format[1]
  - Manual "Licensing End User Guide" in PDF format[1]
  - download links for the HSP tool and the daisychain configuration tool

### 3.1.2 INTECRIO-RLINK License

You require a valid license for INTECRIO-RLINK if you want to use the tool for hardware configuration or the generation of executable code.

To run a virtual prototype with XCP, you need one of the following licenses:

- ETAS Virtual OS Execution Platform (license name: `INCA-VIP`)
- INCA-EIP (license name: `ASCET_INCA-EIP`)

For further information on licensing, see “Licensing the Software” on page 25.

### 3.1.3 System Prerequisites

The system prerequisites are listed in the release notes of INTECRIO-RLINK.

1. PDF reader required

### 3.1.4 Required User Privileges for Installation and Operation

In order to install INTECRIO-RLINK on a computer, you need the user privileges of an administrator. Please contact your system administrator, if necessary.

Using INTECRIO-RLINK does not require user privileges of an administrator.

## 3.2 Installing INTECRIO-RLINK

The following components will be installed during installation:

- INTECRIO-RLINK
  - Hardware Configurator
  - Simulink blocks to access the Hardware Configurator
  - Simulink blocks for the available targets
  - Simulink blocks for hardware signal groups and signals
  - Simulink blocks for real-time OS configuration
  - Simulink blocks for project transfer to INTECRIO and INCA
  - Simulink blocks for stimuli signal configuration

Section 3.2.1 describes the installation of INTECRIO-RLINK, section 3.2.2 describes manual association with MATLAB and Simulink, section 3.2.3 contains special installation features for virtual prototyping, and section 3.2.4 lists the options for command-line installation.

### 3.2.1 Initial Installation

For a list of supported operating systems, see the INTECRIO-RLINK release notes. If you try to install INTECRIO-RLINK on a PC with unsupported OS, an error message opens and the installation is aborted.

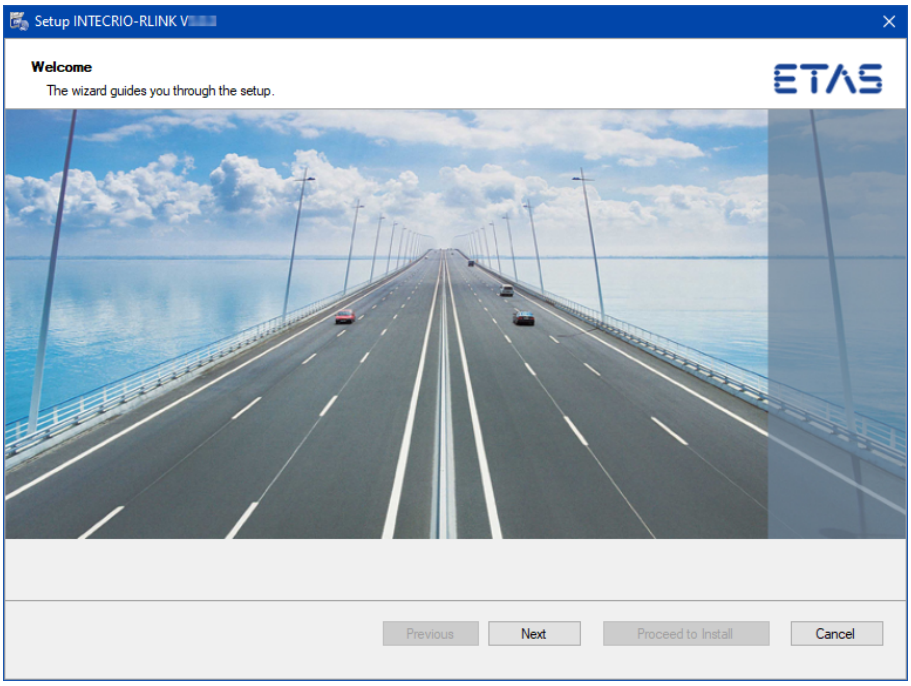
To start the installation

1. Insert the disk in the respective drive on your PC.

   An installation window opens.
2. In that window, follow the "Main" link, then follow the "INTECRIO-RLINK V5.0.*" link.

3. Alternatively, select the drive in the Windows Explorer and run the respective `Setup.exe` file.

   The ETAS Installer is launched.




4. Click **Next** to get to the next installation window.

   Use **Back** to get to the previous window and **Cancel** to cancel installation.

   The installer checks if your computer meets the system requirements for INTECRIO-RLINK installation. The result is displayed in the "System Check" window.

   If your system meets the requirements, the installation continues automatically.

To uninstall a previously installed version

**NOTE**

If no incompatible version of INTECRIO-RLINK is installed on your computer, continue with “License Agreement and Safety Hints” on page 14.

If an INTECRIO-RLINK version incompatible with INTECRIO-RLINK V5.0.2 is present on your computer, that version is listed in the "Uninstall previous products" window.

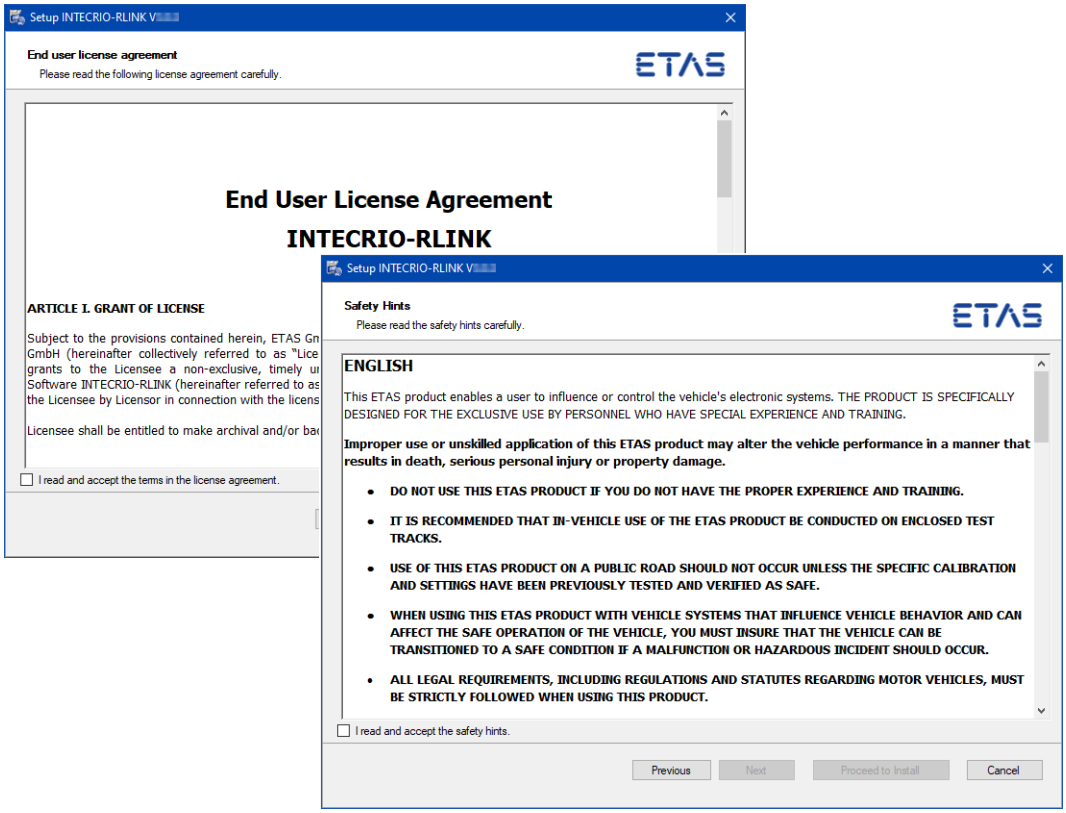
1. Click **Uninstall now**.

   The previous version of INTECRIO-RLINK is uninstalled. You can now continue the installation of INTECRIO-RLINK V5.0.2.

2. When the uninstallation is complete, click **Next** to continue.

License Agreement and Safety Hints

The following two windows show license agreement and safety hints in several languages.




1. Read the license agreement, then activate the **I read and accept the terms in the license agreement** option.

   If you do not accept the license agreement, you cannot continue the installation.

2. Click **Next** to continue.
3. Read the safety hints carefully, then activate the **I read and accept the safety hints** option.

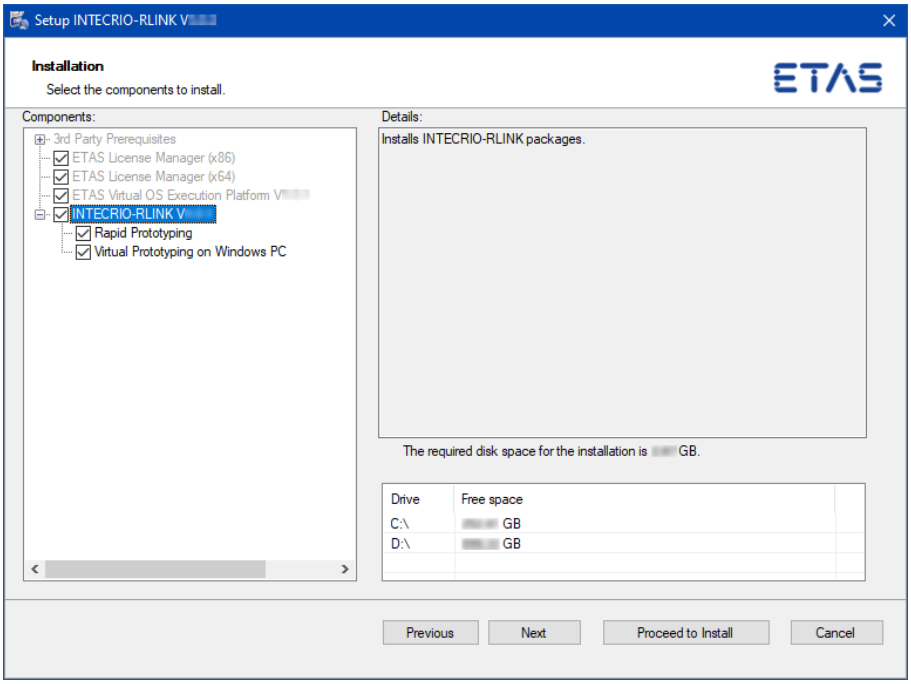
   If you do not accept the safety hints, you cannot continue the installation.

4. Do one of the following:
   - Click **Next** to continue.
   - Click **Proceed to Install** to go to the "Ready to install" window immediately, with default settings for components to be installed, installation paths, etc.

     Continue reading in section "To install INTECRIO-RLINK" on page 17.

To select components

The component selection window lists the necessary 3rd-party prerequisites, ETAS tools (i.e. the license manager and the ETAS Virtual OS Execution Platform) and the components of INTECRIO-RLINK. Prerequisites and components already present on your PC are marked accordingly.




Components whose names appear in black font can be selected or deselected for installation. Thus, you specify the functional scope of the installed software.

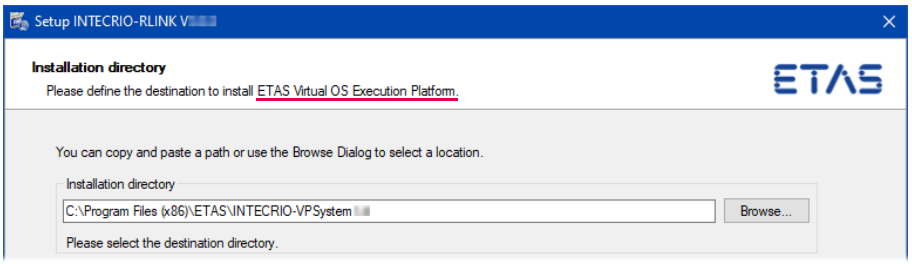
The text field on the right displays information on the highlighted component.

1. To select/deselect a component for installation, activate/ deactivate the respective option.
2. Do one of the following:
   - Click **Next** to continue.
   - Click **Proceed to Install** to go directly to the "Ready to install" window, with default settings for installation paths.

     Continue reading in section "To install INTECRIO-RLINK" on page 17.

To define path settings

In the first "Installation directory" window, you are prompted to enter a destination directory for the ETAS Virtual OS Execution Platform.



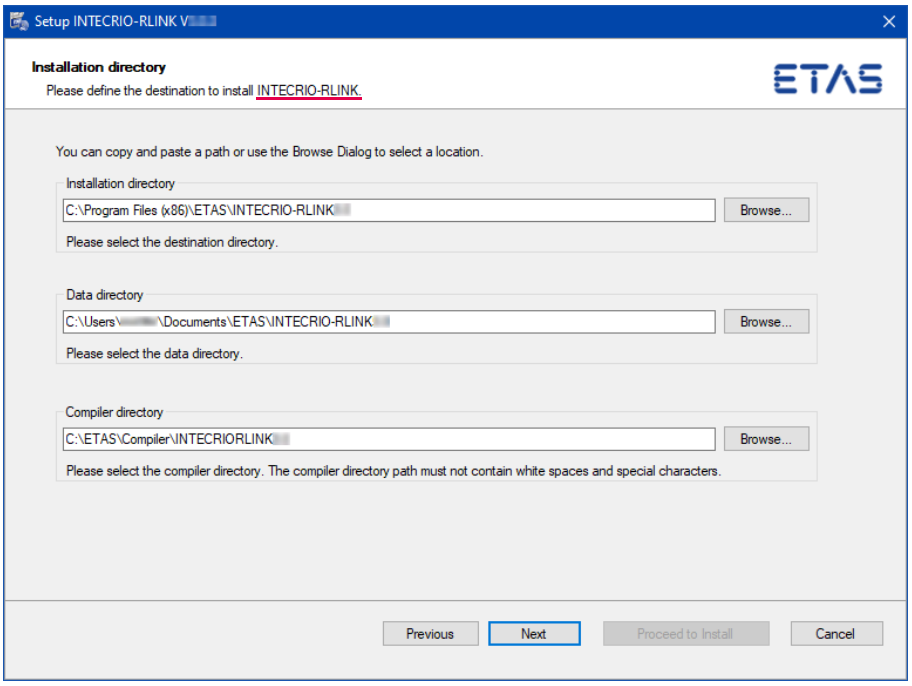

**NOTE**

This window is omitted if the ETAS Virtual OS Execution Platform is already installed on your computer.

1. Enter or select (via the **Browse** button) a valid path.

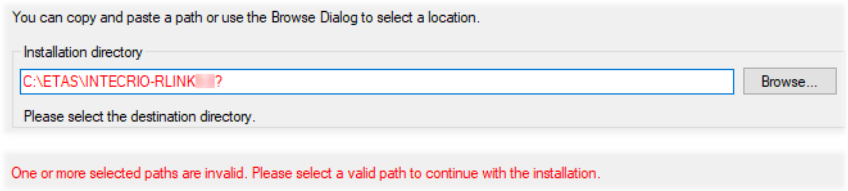
   An invalid path is displayed in red, and a warning appears.
2. Click **Next**.

In the second "Installation directory" window, you are prompted to enter destination directories for the installation (referred to as `<installation>`), for project data files, and for the compilers.



3. To change a preset directory, enter or select (via the **Browse** button) the new path.

   If you entered an invalid path, that path is displayed in red, and a warning appears.



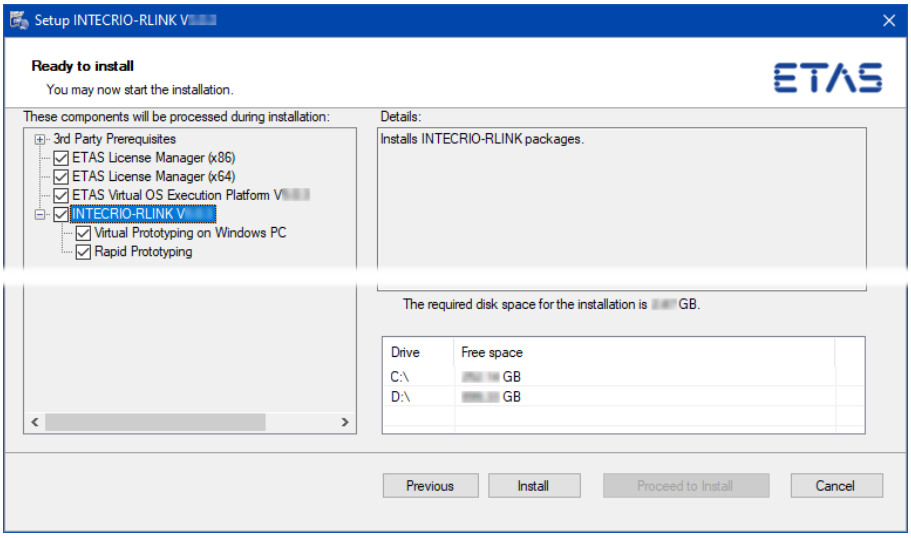

**NOTE**

Paths with blanks are not possible for the compilers used in INTECRIO-RLINK. If you entered a compiler path with blanks, that path is marked as invalid.

4. Click **Next** or **Proceed to Install** to continue.

To install INTECRIO-RLINK

The "Ready to install" window lists the components selected for installation. You cannot change the selection here; to do so, you must go back to the "Installation" window (cf. Page 15).




**NOTE**

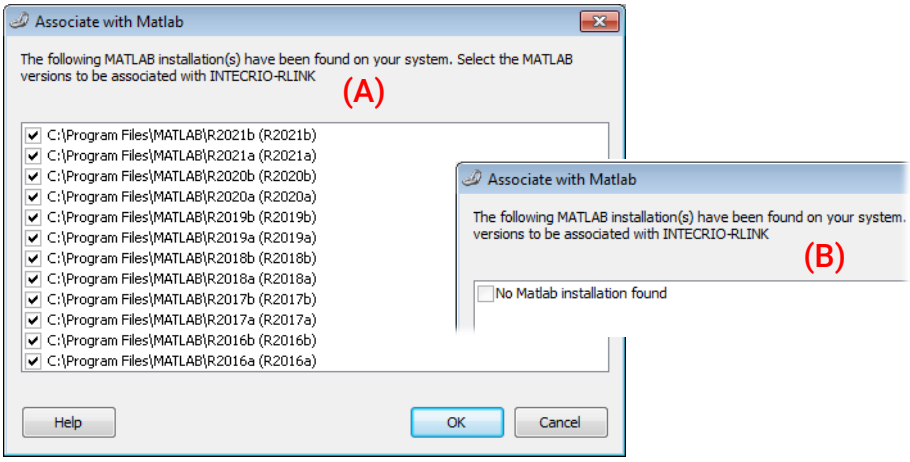
The next step will start the installation. You *cannot* cancel the installation once it is running; the **Cancel** button is deactivated.

1. In the "Ready to install" window, click **Install**.

   The selected components (cf. Page 15) are installed. Components already present on your PC are skipped. A progress indicator shows how the installation is progressing.

To connect INTECRIO-RLINK with MATLAB and Simulink

After the installation is completed, the "Association with MATLAB" window opens. It offers all supported MATLAB and Simulink installations (R2016a – R2021b and their related service packs known at the time of the INTECRIO-RLINK V5.0 release) available on your system for selection (A in the screenshot below). Even if no MATLAB and Simulink installation is found, the "Associate with MATLAB" window opens (B in the screenshot).




**NOTE**

If you want to use a network installation of MATLAB and Simulink, click the **Help** button and follow the instructions given in the message window.

1. In the "Association with Matlab" window, select one of the existing MATLAB and Simulink installations.

   You can select none, one, or several installations.

   **Cancel** closes the window without establishing an association to MATLAB and Simulink.
   INTECRIO-RLINK is installed nevertheless, but you must associate the program to MATLAB and Simulink manually before you can use it. See section 3.2.2 for details.

2. Click **OK**.

   INTECRIO-RLINK is associated with the selected MATLAB and Simulink installation(s).

   You can change the MATLAB and Simulink version associated with INTECRIO-RLINK later; the procedure is described in section 3.2.2.

To complete the installation

Once all components have been installed successfully, you are prompted to end the installation.

1. It is recommended that you activate the **Show readme when setup is closed** option
2. Click **Finish** to end the installation.

In the Windows Start menu, the **ETAS INTECRIO-RLINK 5.0** folder with the following entries is created:

- **Associate to Matlab**

  Connects a selectable MATLAB and Simulink installation with INTECRIO-RLINK (see section 3.2.2).

  This is necessary if MATLAB and Simulink is newly installed without INTECRIO-RLINK being newly installed at the same time.
- **INTECRIO-RLINK Manuals**

  Opens the INTECRIO-RLINK documentation directory.

The ETAS License Manager can be found in the app list of the Windows Start menu at **E > ETAS > ETAS License Manager**.

The ETAS Virtual OS Execution Platform has another separate folder **ETAS Virtual OS Execution Platform *<x>.<y>*** in the Windows Start menu.

- **Signal Configuration Editor V5.0**

  Opens the signal configuration editor where you can configure stimuli signals.

### 3.2.2 Manual Association with MATLAB® and Simulink®

If MATLAB and Simulink were not present during INTECRIO-RLINK installation, or if a new version of MATLAB and Simulink is installed after INTECRIO-RLINK, you must associate INTECRIO-RLINK and MATLAB and Simulink manually.

To associate INTECRIO-RLINK with MATLAB and Simulink manually

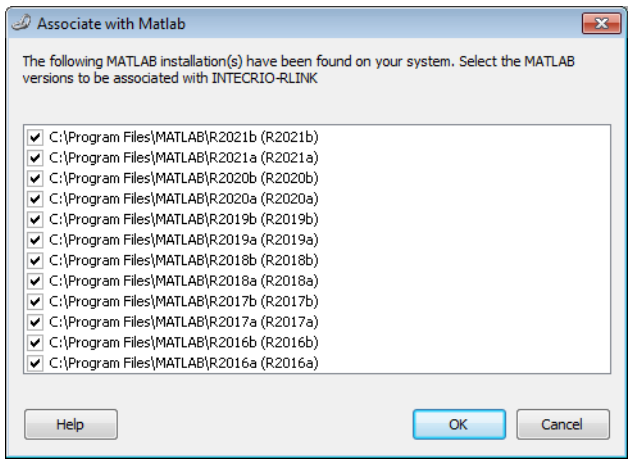
**NOTE**

If you want to use a network installation of MATLAB and Simulink, click the **Help** button and follow the instructions given in the message window.

1. Install MATLAB and Simulink as described in the relevant installation instructions.

2. To associate INTECRIO-RLINK with MATLAB and Simulink, open the Windows Start menu, go to the `INTECRIO-RLINK 5.0` folder and select **Associate to Matlab**.

   The "Associate with Matlab" window opens. It offers all supported MATLAB and Simulink installations available on your system for selection.




3. In the "Associate with Matlab" window, select one or more MATLAB and Simulink installations.
4. Click **OK** to continue.

   INTECRIO-RLINK and the selected MATLAB and Simulink installation are connected.

### 3.2.3 Prerequisites for Virtual Prototyping on a Windows PC

Virtual prototyping with INTECRIO-RLINK requires RTA-OSEK Tools and RTA-OSEK for PC. Since INTECRIO-RLINK V1.4, these products are installed with INTECRIO-RLINK as parts of the *ETAS Virtual OS Execution Platform* installation (see “To select components” on page 15).

### 3.2.4 Command-Line Installation

When you start the INTECRIO-RLINK installation from a command line, you can use several command-line parameters to customize the installation.

Each execution of `Setup.exe` writes a log file *<date_time>* `Setup.log`. These log files are stored in the `C:\ProgramData\ETAS\SETUP\Logs` folder.

**/silent**

*Silent installation* mode. With this installation mode, no dialog windows requiring user information open.

> **NOTE**
>
> To accept EULA and safety hints (cf. Page 14) during silent installation, you must use the command-line parameters **/EULAAccepted** and **/SafetyHintsAccepted**.
>
> To deal with a possible request for a computer restart, you must use either the **/NoRestart** or the **/AllowRestart** command-line parameter.

Default values are used for all information normally requested in installation windows. Error messages are hidden, too.

**/EULAAccepted**

Accepts the license agreement.

The text of the license agreement is provided on the installation disk, in the `EULA` subfolders.

**/SafetyHintsAccepted**

Accepts the safety hints.

The text of the safety hints is provided on the installation disk, in the `SafetyHints` subfolders.

**/NoRestart**

Suppresses a computer restart that may be required at the end of the installation. If a reboot is suppressed, a log message is issued.

**/AllowRestart**

Allows a computer reboot restart that may be required at the end of the installation. A restart is performed without further notice.

**/UninstallPreviousVersion**

Uninstalls an existing older INTECRIO-RLINK version installed on your computer.

> **NOTE**
>
> If you do not use this command-line parameter, `setup.exe` will abort with an error If a previous INTECRIO-RLINK version is found.

`/UninstallPreviousVersion` is not used in combination with `/Uninstall`.

**/Debug**

Writes additional log files for `*.msi` packages.

These files are stored in the `C:\ProgramData\ETAS\SETUP\Logs` folder.

**/DefaultSettings**

Allows to specify an own XML file with default settings (instead of using `InstallationDefaultSettings.xml`), e.g., the following:

| Variable | Meaning | See also |
|---|---|---|
| `PRODINSTDIR` | installation directory | Page 16 |
| `PRODUSERDOCUMENTSDIR` | data directory | |
| `COMPILERDIR` | compiler directory | |

You can specify a relative path if the file is relative to `Setup.exe`, otherwise you have to specify an absolute path.

Syntax: `/DefaultSettings:"<path>\<filename>.xml"`

**/Uninstall**

Uninstalls INTECRIO-RLINK. Can be combined with `/silent` for uninstallation without user interaction.

**/Repair**

In combination with `/Silent`, the repair process is triggered. Otherwise, the maintenance mode is triggered.

**NOTE**

`/Uninstall` and `/Repair` cannot be used with `setup.exe` in the installation location. You ***must*** use the `setup.exe` file provided in the `C:\<programs>\ETAS\GENericSetup\IPE INTECRIO-RLINK\5.0.<x>.<y>` folder.

*<programs>* = `Program Files (x86)` (64 bit OS) or `Program Files` (32 bit OS)

### *Examples*

`Setup.exe /silent /EULAAccepted /SafetyHintsAccepted`

Triggers a silent installation.

`"C:\Program Files (x86)\ETAS\GENericSetup\IPE INTECRIO-RLINK\5.0.<x>.<y>\Setup.exe" /uninstall /Debug`

Triggers a non-silent uninstallation and writes additional logs.

`Setup.exe /DefaultSettings:”D:\myOwnSettings.xml”`

Triggers a non-silent installation that uses your own default settings for the installation.

## 3.3 Setting the Licensing Behavior

The INTECRIO-RLINK installation includes an installation of the ETAS license manager. You can define the way in which INTECRIO-RLINK (and other ETAS software programs) accesses the required licenses in the `[Licensing]` section of an `*.ini` file.

The folder `Installation\INTECRIO-RLINK\Packages\Blockset INTECRIO-RLINK_Licensing*` on your installation disk contains such an `*.ini` file, i.e. `Licensing.ini`.

## General Procedure

The procedure to edit the `Licensing.ini` file and start the installation depends on the original location of the installation files.

**The installation files are placed in a directory where you cannot edit them:** Use one of the following procedures.

- Copy *all* installation files to a directory where you can edit the files.

  There, edit `Licensing.ini` (cf. Page 23), then start the installation via double-click `Autostart.exe` or `Setup.exe` (cf. section 3.2.1) or via command line (cf. section 3.2.4).
- Copy only `Licensing.ini` to a directory where you can edit the file.

  There, edit `Licensing.ini` (cf. Page 23). Create your own XML file with default settings (cf. Page 22) and enter the location of your edited `Licensing.ini` file in the `LIMA_INFILE` variable:

  ```
  <Variable
       name="LIMA_INIFILE"
       defaultValue="<path>\Licensing.ini"
       validation="fileExists" />
  ```

  Start the installation via command line (cf. section 3.2.4). and use `/DefaultSettings:"<path>\<filename>.xml"` to add your own XML file and with that your `License.ini` file.

**The installation files are placed in a directory where you can edit them:** Edit the `Licensing.ini` file (cf. Page 23) and start the installation.

## Editing the `Licensing.ini` File

To define the access to the required licenses

1. Open the `Licensing.ini` file with a text editor.
2. Modify the settings in the `[Licensing]` section as desired.

   The parameters and their settings are described below.
3. Save your changes.

The following parameters may be used:

- `LicenseFileName`

  Defines the absolute path and file name of the license file which is to be added.

- `LicensesToBorrow`

  You can use this setting if licenses can be borrowed from a license server. To enable the borrow mechanism, you must enter the name of the product or features license (e.g., INTECRIO-RLINK). If you enter more than one license, the license names must be separated by blanks.

  INTECRIO-RLINK uses the following licenses:

| License name | Functionality |
|---|---|
| `BLOCKSET_ES900` | Target connector for ES800 and ES900 |
| `BLOCKSET_PC` | Target connector for virtual prototyping |
| `BLOCKSET` | Base functionality |
| `INCA-VIP` | ETAS virtual OS execution platform |

- `BorrowExpiryMode`

  Defines the way in which the expiration of the borrow status is given. Possible values are:

  - `Date`

    If the `BorrowExpiryMode` is set to `Date`, the borrow period will expire at a certain date which is specified under `BorrowExpiry-Date`.

  - `Interval`

    If the `BorrowExpiryMode` is set to `Interval`, the borrow period will expire after a certain number of days which is specified under `BorrowExpiryInterval`.

- `BorrowExpiryDate`

  If the `BorrowExpiryMode` is set to `Date`, this parameter specifies the date when the borrow period expires. The format is `yyyy-mm-dd`.

- `BorrowExpiryInterval`

  If the `BorrowExpiryMode` is set to `Interval`, this parameter specifies the length of the borrow period in days.

- `ExecuteBorrowAutomaticExtensionInterval`

  Defines at what point of time the borrow period will be automatically extended. This parameter specifies the number of days before the expiration of the current borrow period. When this time is reached, the borrow period is automatically extended to the interval specified under `BorrowAutomaticExtensionInterval`.

- `BorrowAutomaticExtensionInterval`

  This parameter specifies the borrow interval in days that is applied when an automatic extension of the borrow period is executed (as defined under `ExecuteBorrowAutomaticExtensionInterval`).

- `ImmediateBorrow`

  You can define that a license is automatically borrowed. Possible values are:

- `True`
  The license is borrowed automatically at installation time.

  > **NOTE**
  >
  > `ImmediateBorrow='True'` works **only** for the user who performs the installation. Other users who work on the same computer do not own the borrowed license.

- `False`
  The license will be borrowed at the first time when the program connects to the license server.

- `CustomLicenseFolder`
  Due to the fact that the default location for added license files (e.g., `C:\Program Data\ETAS\FlexNet`) is only writable for users with admin rights, a different path for the license file folder may be specified with this parameter.

The following example defines that borrowing is enabled for all virtual-prototyping parts. The licenses will be borrowed when INTECRIO-RLINK is started for the first time; by default the licenses expire after 100 days.

```
[Licensing]
LicenseFileName = 'd:\licenses\MyLicense.lic'
LicensesToBorrow = 'BLOCKSET BLOCKSET_PC'
BorrowExpiryMode = 'Interval'
BorrowExpiryInterval = '100'
ImmediateBorrow = 'false'
```

## 3.4 Licensing the Software

A valid license is required for using INTECRIO-RLINK. You can obtain the license file required for licensing either from your tool coordinator or through a self service portal on the ETAS Internet Site under https://www.etas.com/support/licensing. To request the license file you have to enter the activation number which you received from ETAS during the ordering process.

In the Windows Start menu, go to the app list and select **E > ETAS > ETAS License Manager**.

Follow the instructions given in the dialog. For further information about, for example, the ETAS license models and borrowing a license, press <F1> in the ETAS License Manager.

## 3.5 Uninstalling INTECRIO-RLINK

If you uninstall INTECRIO-RLINK, all components are uninstalled automatically. The ETAS Virtual OS Execution Platform

Use one of the following ways to start the uninstall process:

A **Programs and Features** in the Windows Control Panel

B the `/Uninstall` command-line parameter (Page 22)

To uninstall INTECRIO-RLINK

1. Start the uninstall procedure.

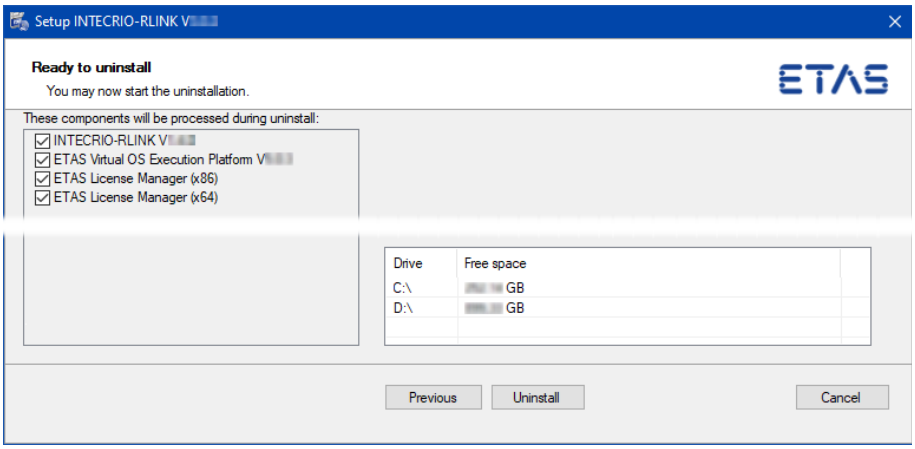
   The ETAS installer is launched.

   You can use the **Next** button to get to the next uninstallation window; **Back** to get to the previous window, and **Cancel** to cancel uninstallation.

2. Click **Next** to check if your computer meets the system requirements for INTECRIO-RLINK uninstallation.

   The result is displayed in the "System Check" window.

   If the system requirements for uninstallation are met, the "Ready to uninstall" window opens. It lists the components that will be uninstalled. You cannot change the selection.




   The ETAS Virtual OS Execution Platform and the License Manager are only uninstalled if no other program that uses them is available on your PC.

**NOTE**

The next step will start the uninstallation. You *cannot* cancel uninstallation once it is running; the **Cancel** button is deactivated.

3. Click **Uninstall** to actually uninstall INTECRIO-RLINK

   A progress indicator shows how the uninstallation is progressing.

   Once all components have been uninstalled, a success window opens.

4. Click **Finish** to end the uninstallation.

# 4 INTECRIO-RLINK Quick Guide

## 4.1 Introduction

This chapter is intended to provide a quick guide to the most important features of INTECRIO-RLINK V5.0. The examples, displayed in the form of flowcharts, provide you with an overview of the program functionality and operating mode. In these flowcharts, dashed black arrows lead from an action to its result. Solid red arrows mark locations where actions shall take place.

**NOTE**

The virtual prototyping (VP) features described in this chapter are only available if you installed the **Virtual Prototyping on Windows PC** component (cf. Page 15).

This chapter does *not* explain how to work with Simulink in general.

Below, you find a short explanation of the most important abbreviations used here.

- **DB** – Database
- **EE** – ETAS Experiment Environment
- **HBB** – Hook-based bypass
- **HC** – Hardware Configurator
- **HW** – Hardware
- **HWS** – Hardware System
- **ISR** – Interrupt service routine
- **OS** – Operating system
- **RP** – Rapid prototyping
- **SBB** – Service-based bypass
- **VP** – Virtual Prototyping
- **WS** – Workspace

## 4.2 Preparing the Model

INTECRIO-RLINK models require a special system target and a fixed-step solver.

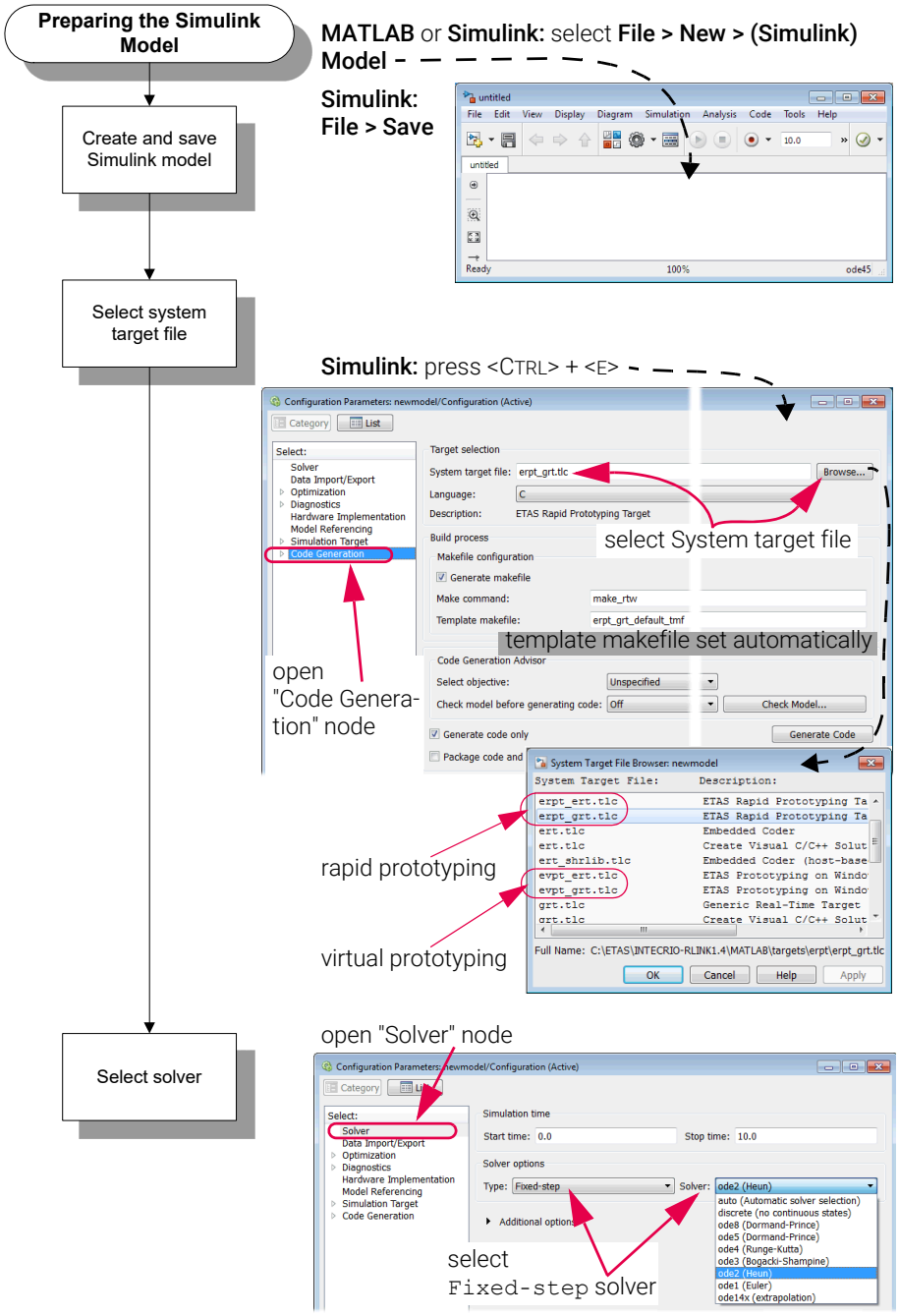



**Fig. 4-1** Preparing the Simulink Model

## 4.3 Creating and Configuring a Hardware System

You can add a hardware system to a new Simulink model or to an existing model. Before you can open the Hardware Configurator, you must save a new model so that its path is determined.

Use the Hardware Configurator to configure the hardware system. The "Insert New Item" window contains the elements available on every hierarchical level.

You can make various settings (e.g., channel settings for measure hardware etc.) for every hardware component using the relevant dialogs.

The `Hardware Systems` folder is the folder for hardware systems (HWS) in the workspace.

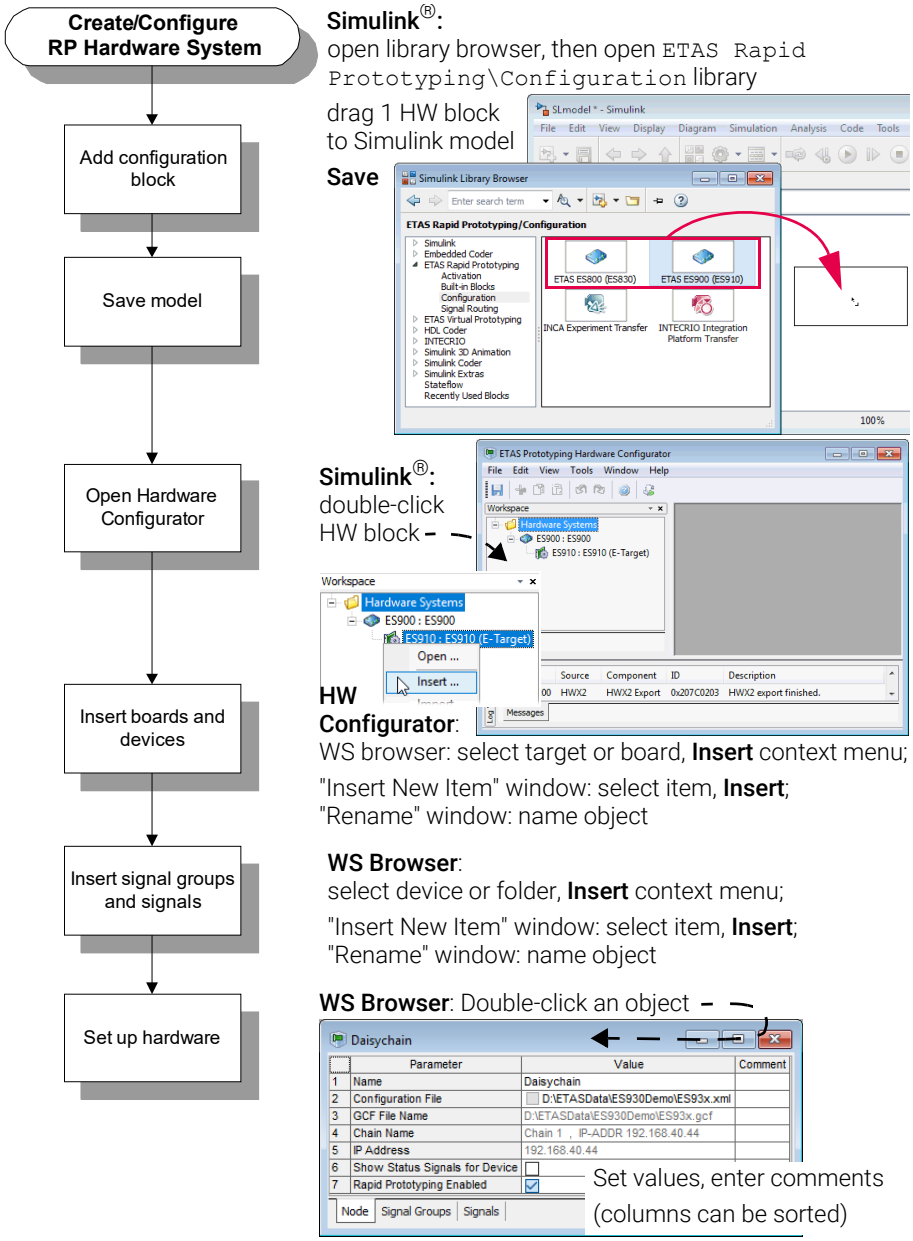



**Fig. 4-2** Creating and Configuring an RP Hardware Configuration

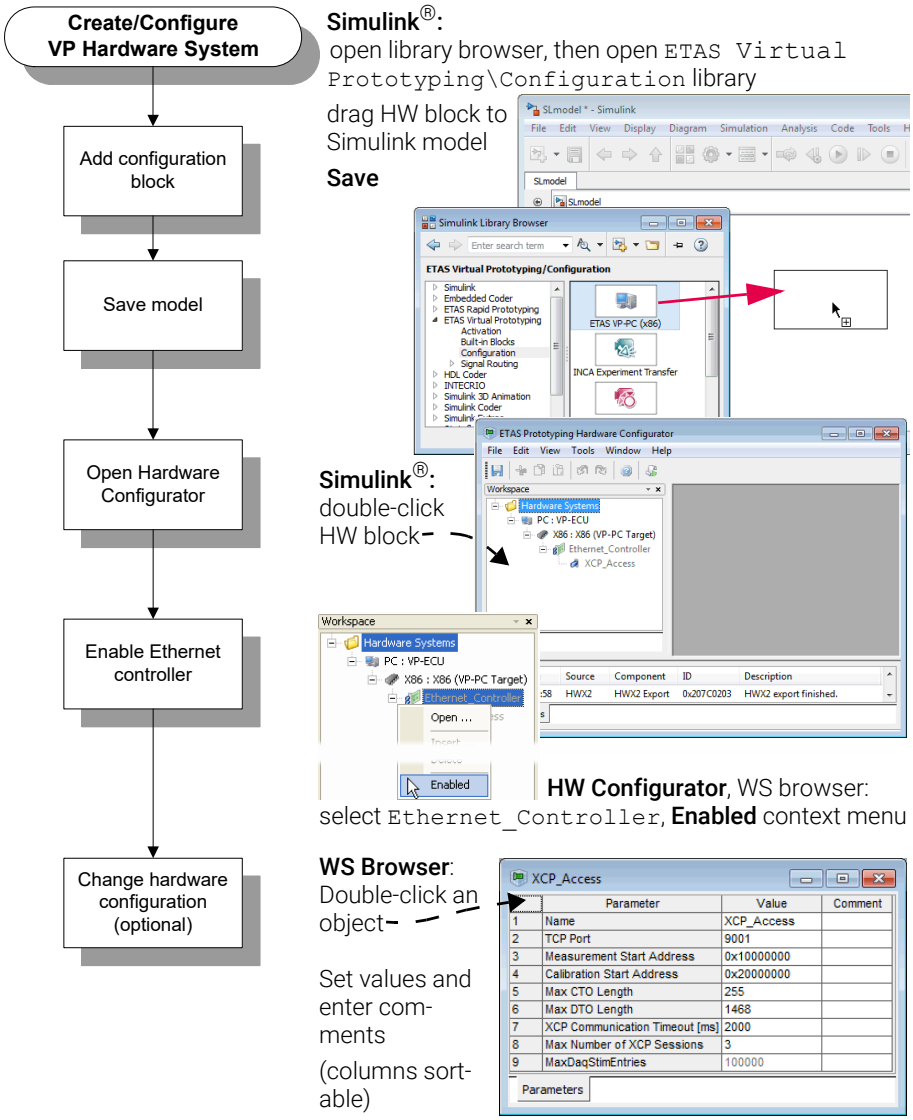



**Fig. 4-3** Creating and Configuring a VP Hardware Configuration

### 4.3.1 Importing a Hardware System

As an alternative to manual creation and setup of a HWS, you can import a hardware description, `*.hwx`, created with INTECRIO, INTECRIO-RLINK or ASCET-RP into a new, or an existing, hardware system.

The file extension `*.hwx` is used for two description formats (HWX1 and HWX2).

> **NOTE**
>
> Since INTECRIO-RLINK V1.5.3, the HWX1 format can no longer be imported.

After import file selection, HWX2 import is aided by a wizard (see Fig. 4-4).

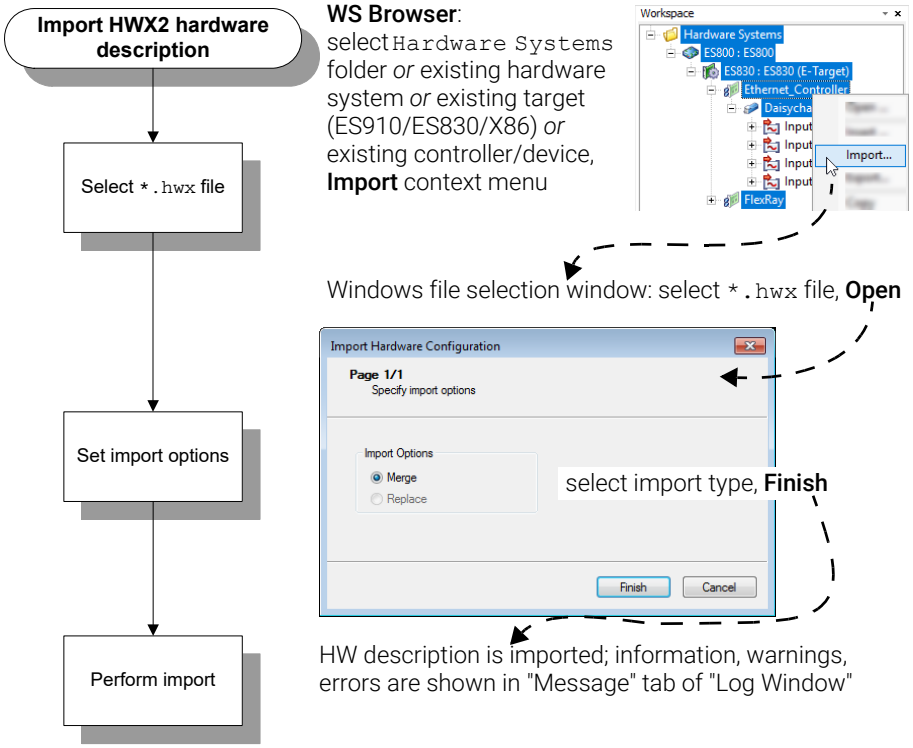



**Fig. 4-4** Importing a HWX2 hardware description (`*.hwx` file)

## 4.3.2 Configuring a Daisychain

The configuration of a daisychain follows a different procedure.

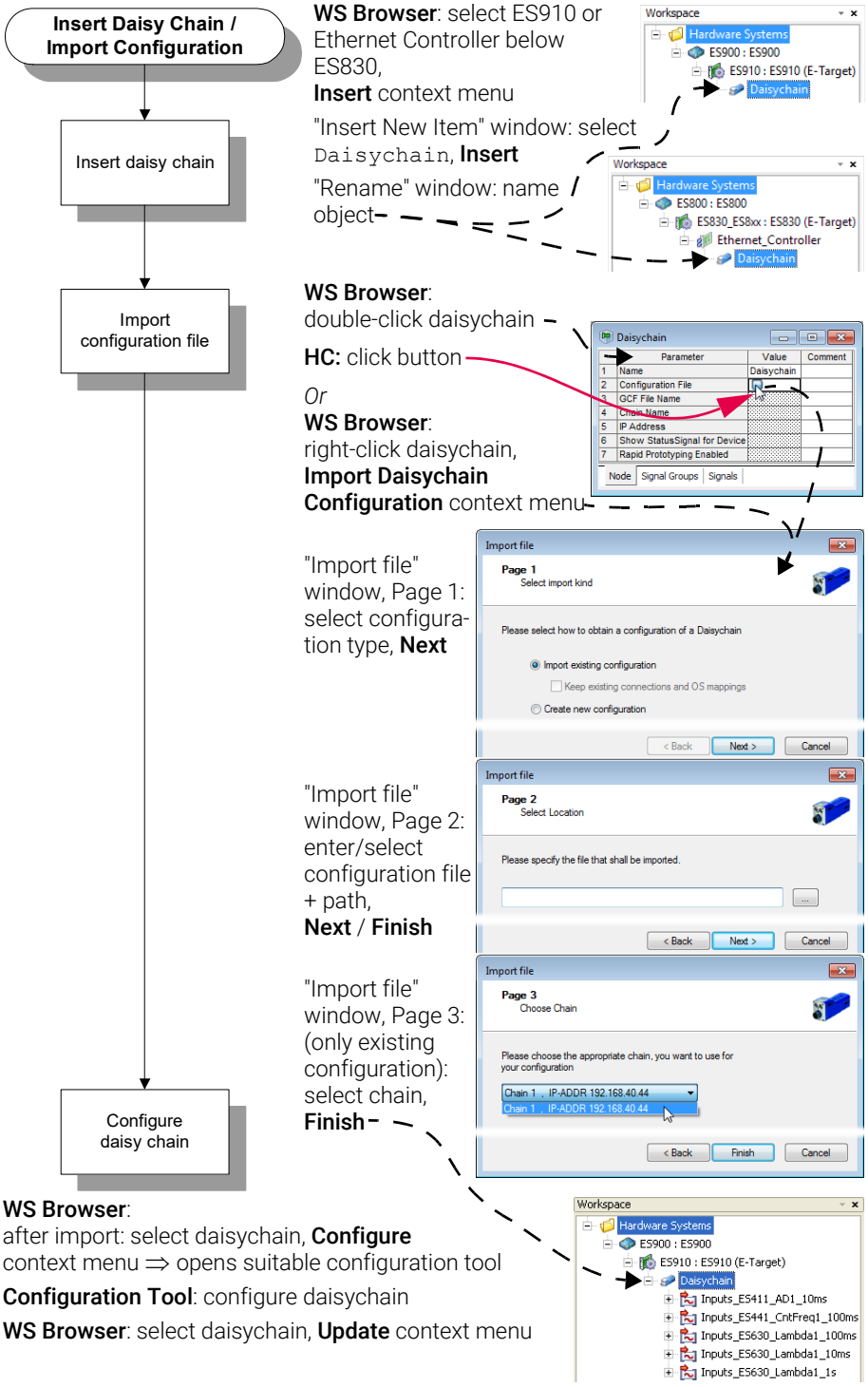



**Fig. 4-5** Creating and Configuring a Daisychain

## 4.3.3 Configuring a LIN Controller

The configuration of a LIN controller can be imported.

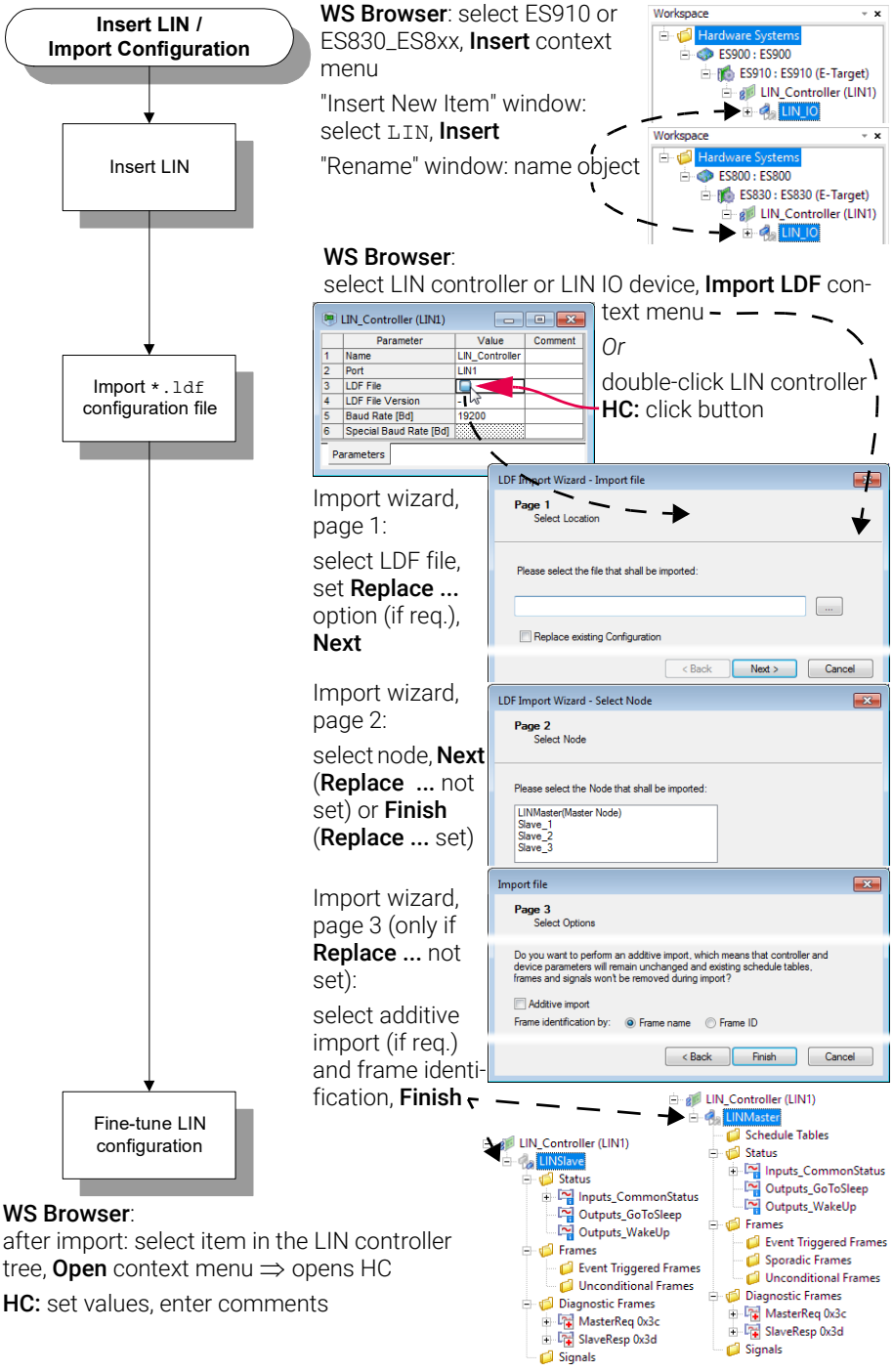



**Fig. 4-6** Creating and Configuring a LIN Controller

LIN nodes (devices) can be of type Master or Slave. You can switch the node type; type-dependent settings are adjusted automatically.

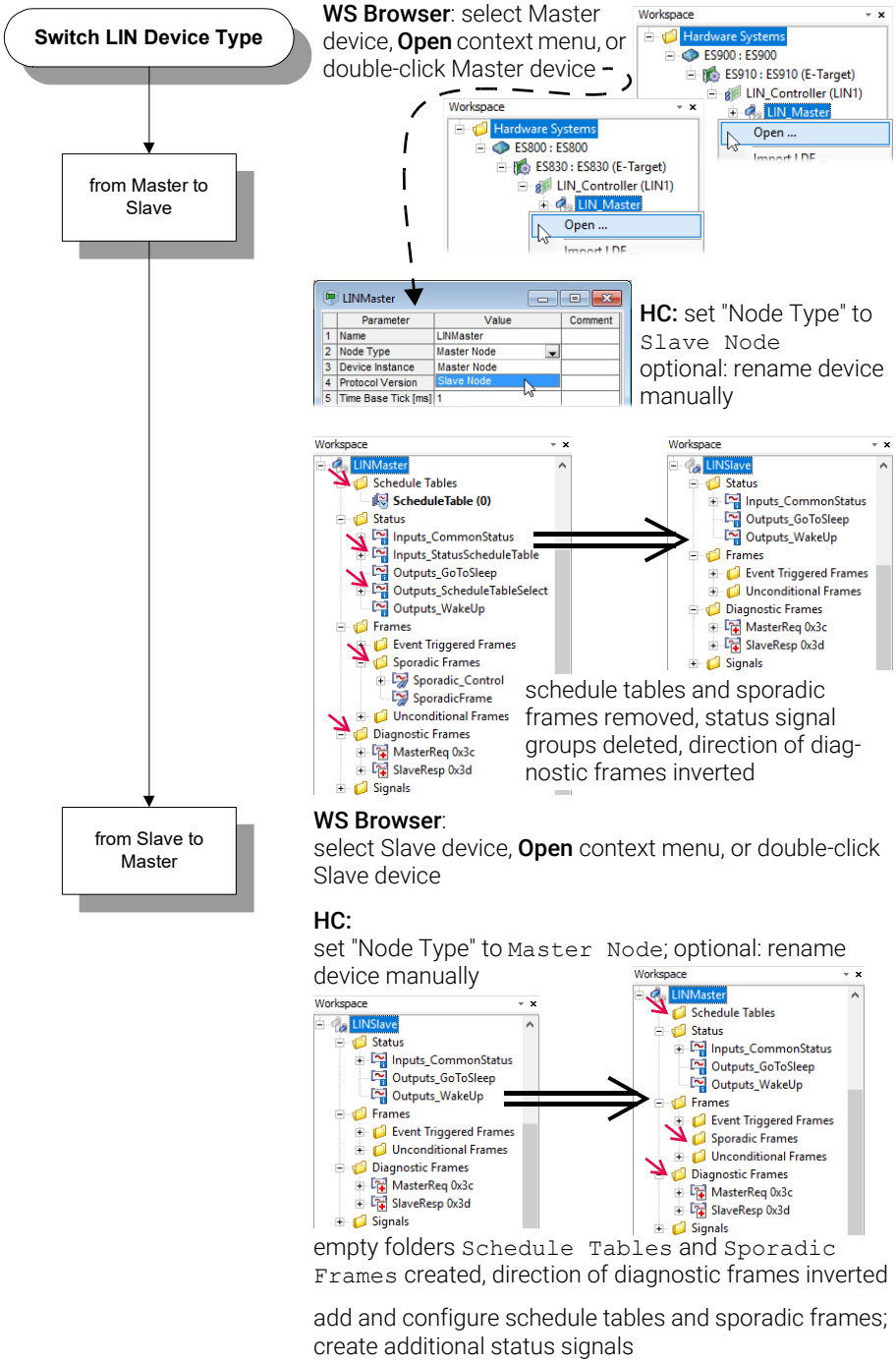



**Fig. 4-7** Switching the LIN Node Type

## 4.3.4 Configuring a Bypass

> **NOTE**
>
> Service-based bypass on an ETK with 8 Mbit/s is not supported.
>
> Since INTECRIO-RLINK V5.0.1, FETK bypass on ES910 is deprecated.

After the bypass device is added (see Fig. 4-3), a bypass is configured as follows.

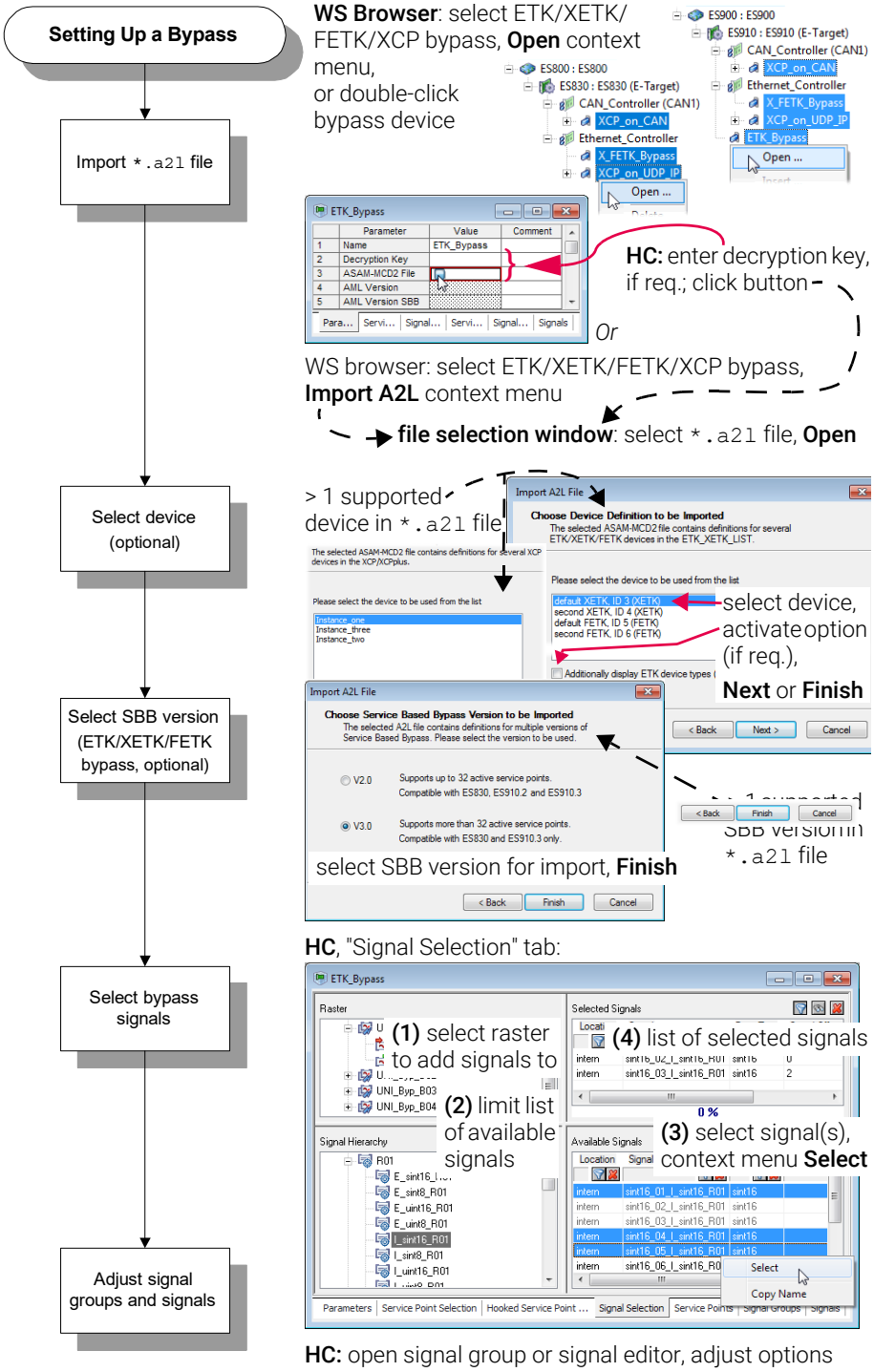



**Fig. 4-8** Setting up a Bypass Device

The service point descriptions for a service-based bypass are provided in the `*.a2l` file. The user selects the actually needed service points and signals after importing the `*.a2l` file.

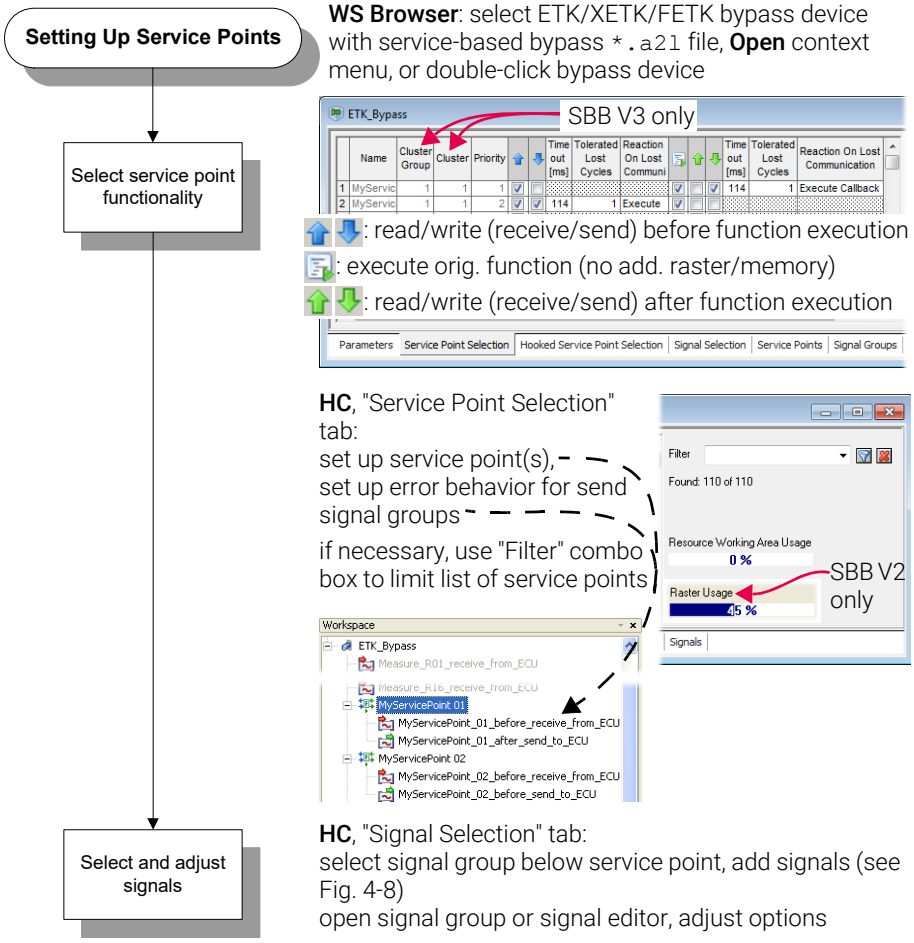



**Fig. 4-9** Setting up Service Points (Service-Based Bypass)

> **NOTE**
>
> Service-based bypass on an ETK with 8 Mbit/s is not supported.

In connection with Distab17, the concept of service point configuration extends to the hook-based bypass. You have to set up hooked service points before you can select the signals.

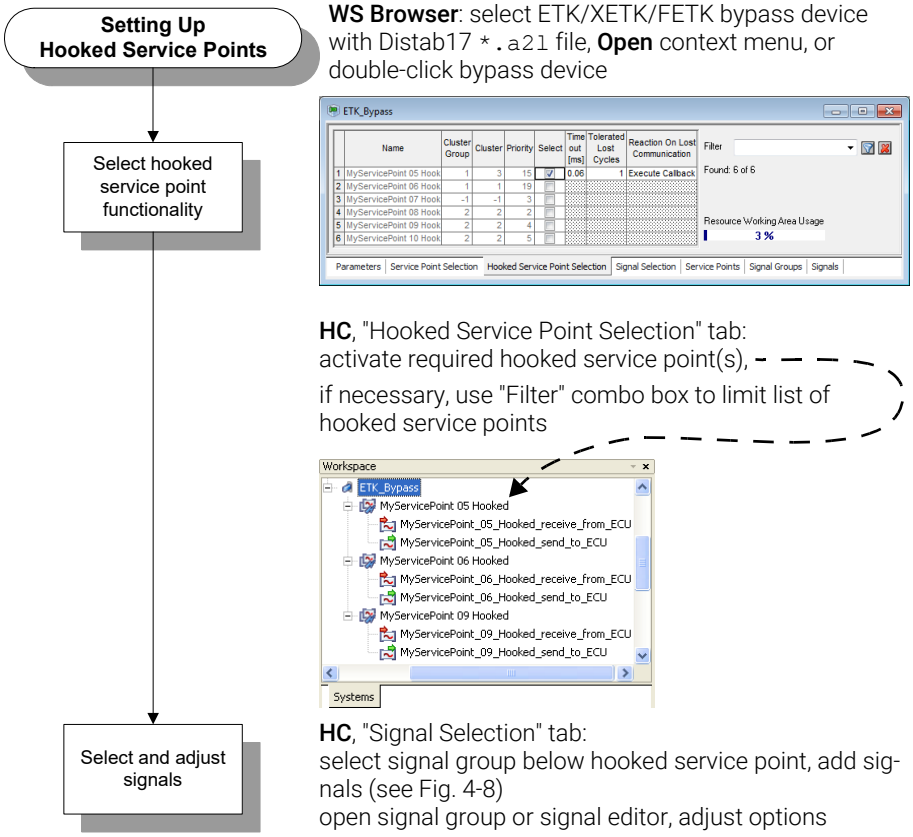



**Fig. 4-10** Setting up Hooked Service Points (Hook-Based Bypass + Distab17)

## 4.3.5 Importing a CAN Configuration File

After a CAN device is added (see Fig. 4-3), you can create signals and frames manually, or you can import a CAN Database. Both classic CAN and CAN FD (ES910+ES922 and ES830) are supported.

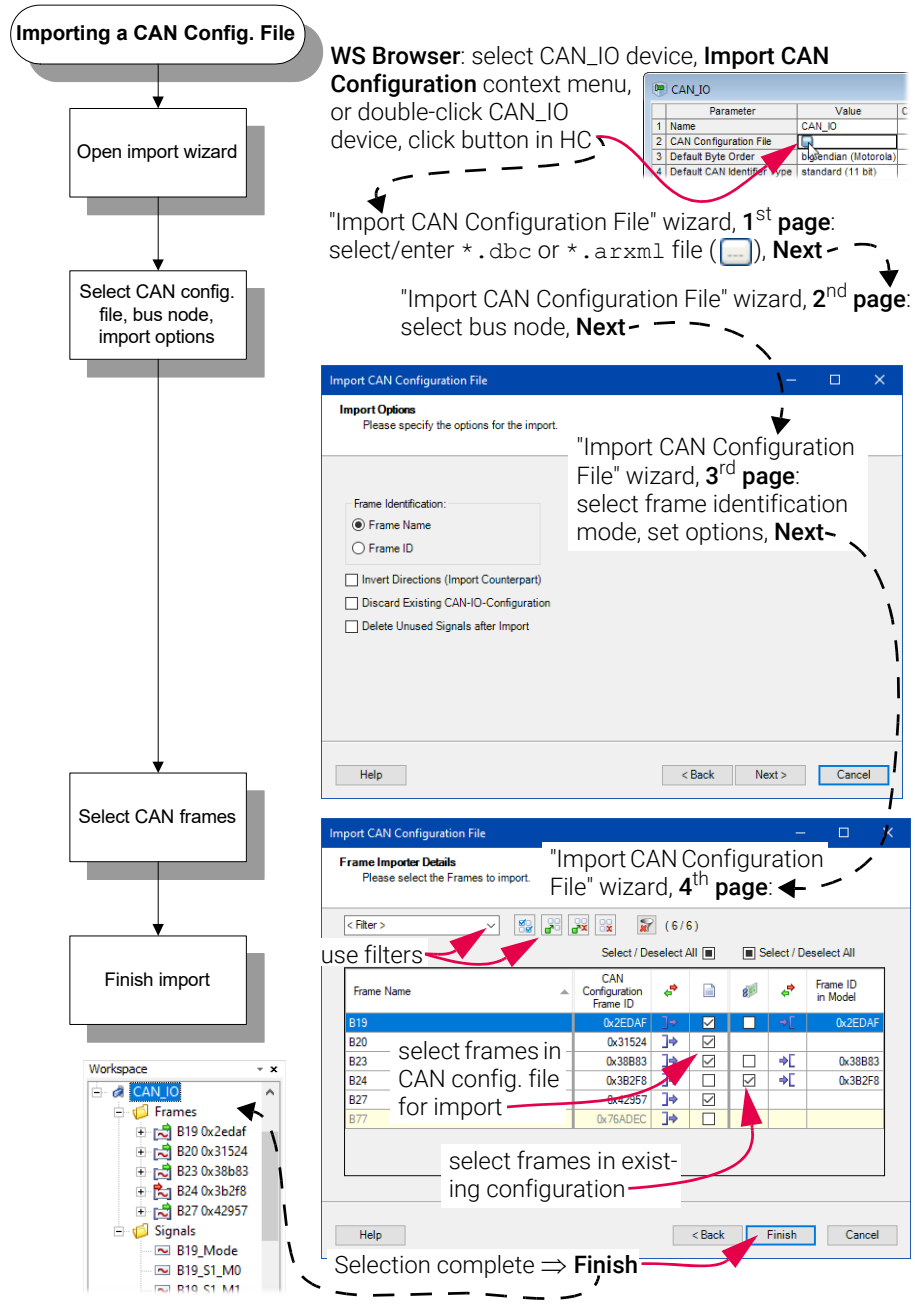
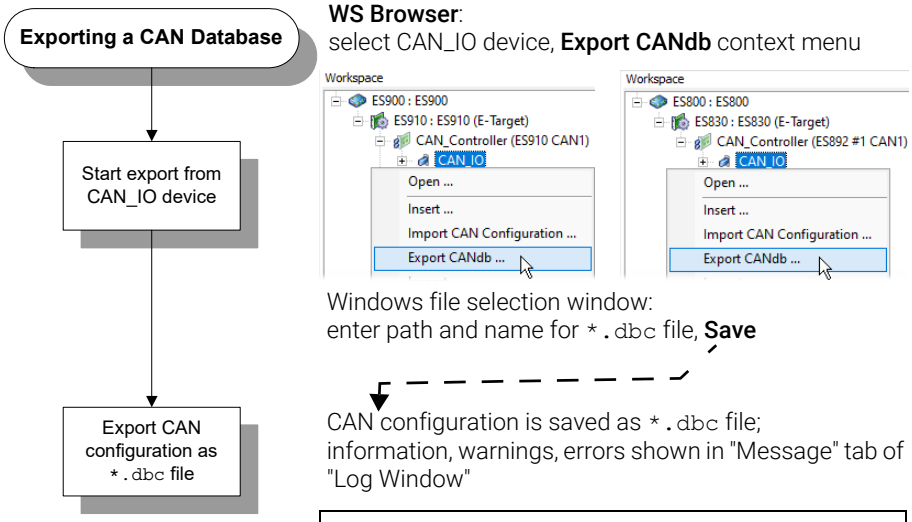



**Fig. 4-11** Importing a CAN Configuration File

### 4.3.6 Exporting a CAN Database

> **NOTE**
>
> Exporting a CAN configuration as `*.arxml` file is currently not supported.

After you have configured a CAN device, you can export the configuration as CAN database file (`*.dbc`). Both classic CAN and CAN FD (ES910+ES922 and ES830) are supported.




> **NOTE**
>
> Some items in CAN config. cannot be exported:
> - empty receive frames / multiplex groups
> - HC parameters not available in CANdb format

**Fig. 4-12** Exporting a CAN Database

## 4.4 Specifying the Model

This section focuses on how to use the blocks and features provided by INTECRIO-RLINK.

For information on standard MATLAB and Simulink functionality, refer to the MATLAB and Simulink documentation.

## 4.4.1 Adding Hardware Signal Groups and Signals

The blocks for adding hardware signal groups and signals are provided in the "ETAS Rapid Prototyping" library, "Signal Routing Blocks" sub-library.

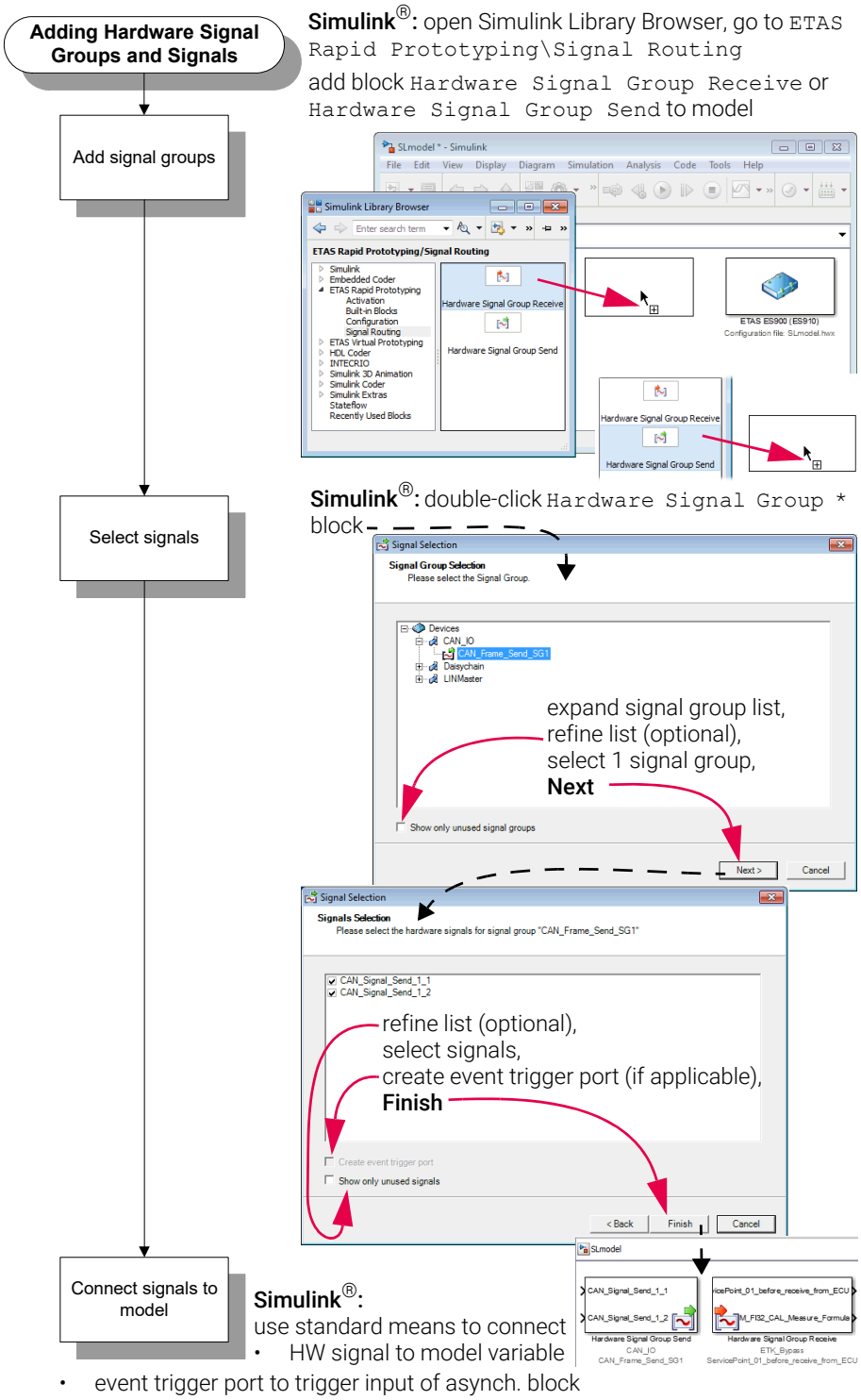



**Fig. 4-13** Adding hardware signal groups and signals

## 4.4.2 Adding Activation Task Blocks

The activation task blocks are provided in the "ETAS Rapid Prototyping" and "ETAS Virtual Prototyping" library, "Activation" sub-library.

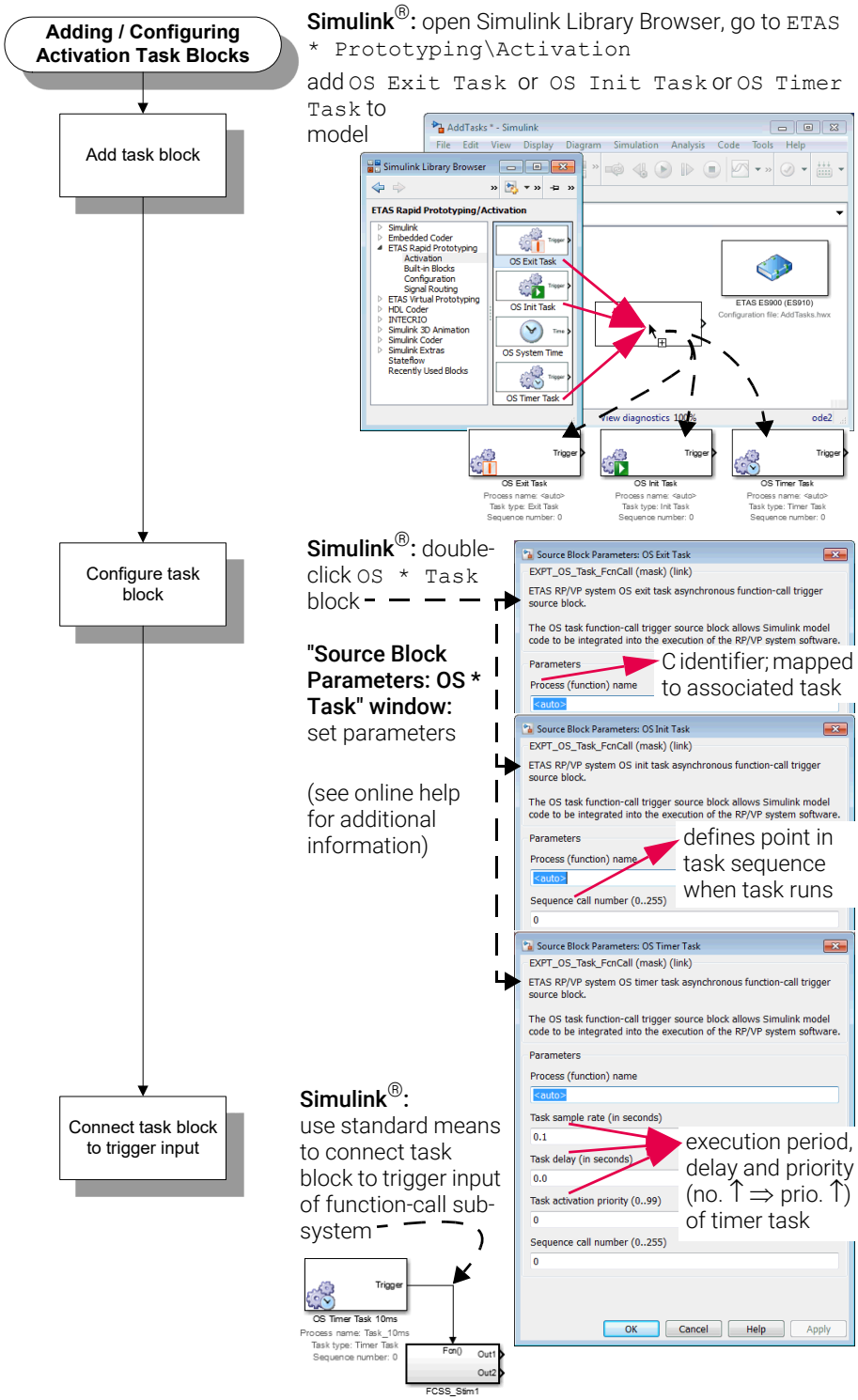



**Fig. 4-14** Adding and configuring activation task blocks

### 4.4.3 Adding the System Time Block

The OS System Time block provides a data output port for the target system time to be read. The block is provided in the "ETAS Rapid Prototyping" and "ETAS Virtual Prototyping" library, "Activation" sub-library.

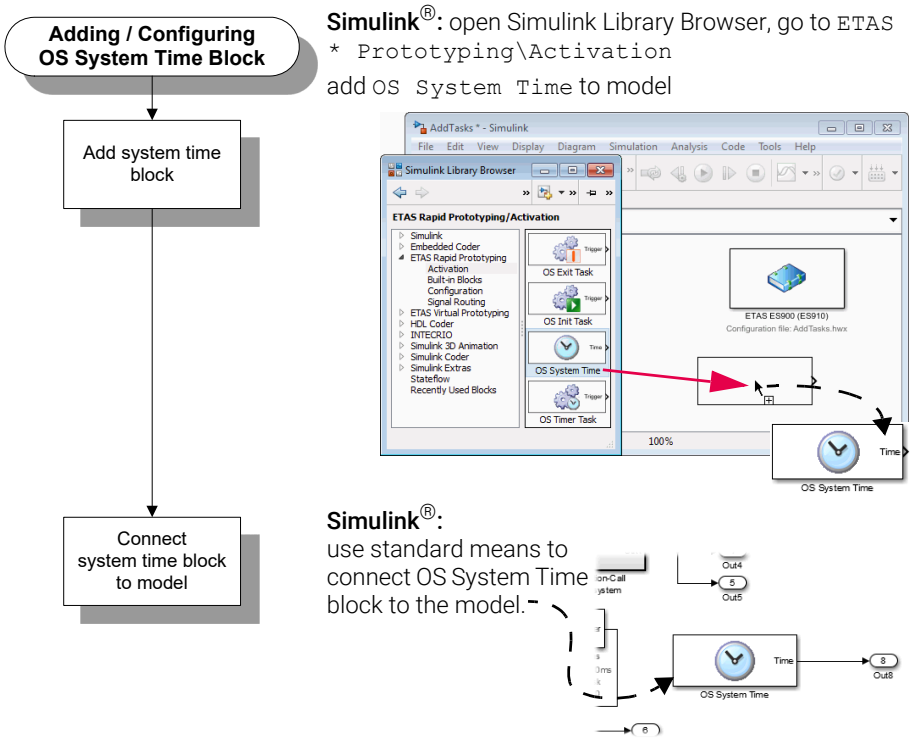



**Fig. 4-15** Adding and configuring an OS System Time block

## 4.5 Stimuli Signal Configuration

> **NOTE**
>
> This section is only relevant if you installed the **Virtual Prototyping on Windows PC** component (cf. Page 15).

The Virtual Prototyping on Windows PC component enables you to define a stimuli signal configuration for use with your controller function.

## 4.5.1 Stimuli Signal Configuration Block

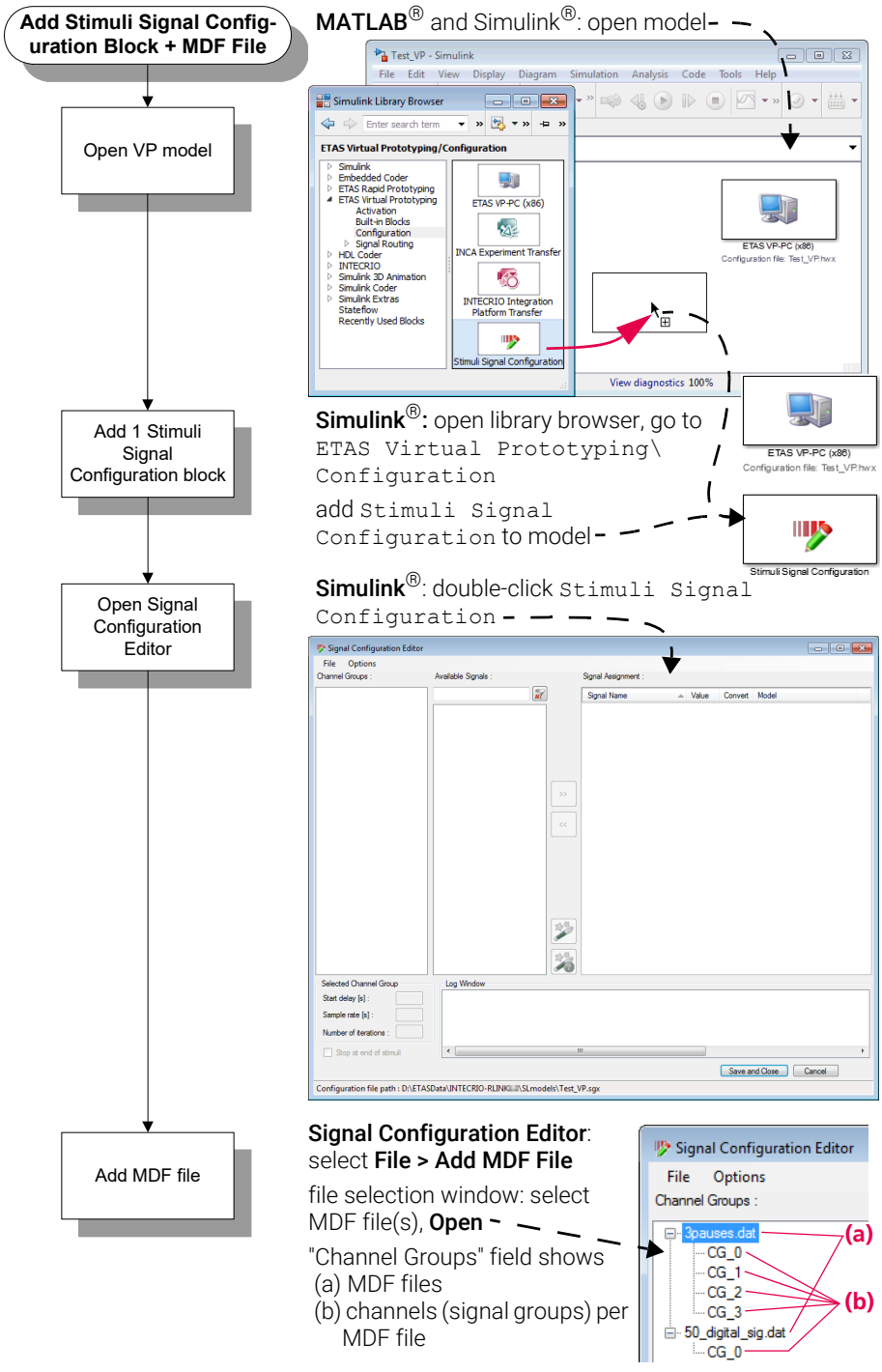



**Fig. 4-16** Adding a Signal Configuration block and MDF file(s)

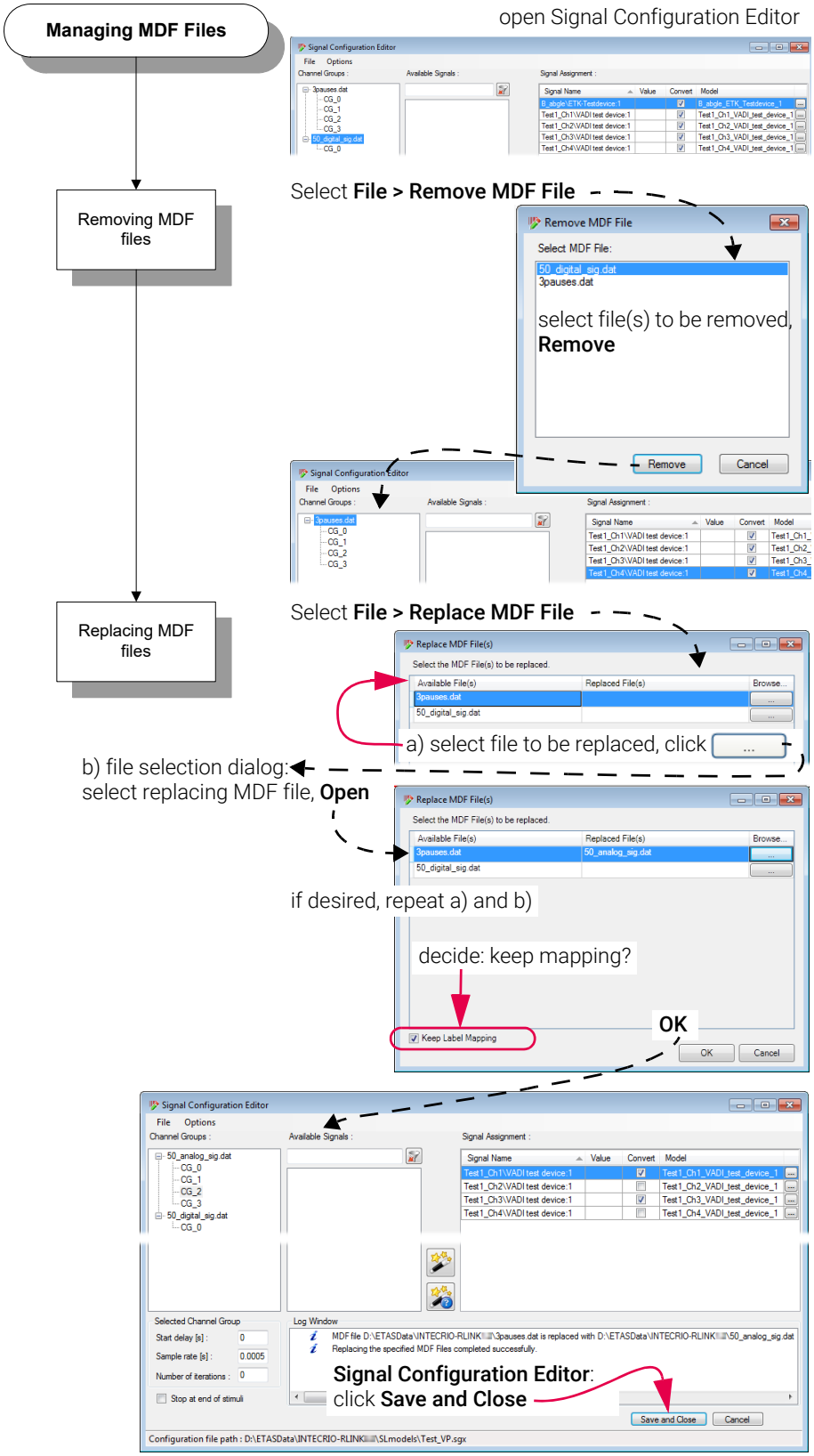



**Fig. 4-17** Managing MDF files

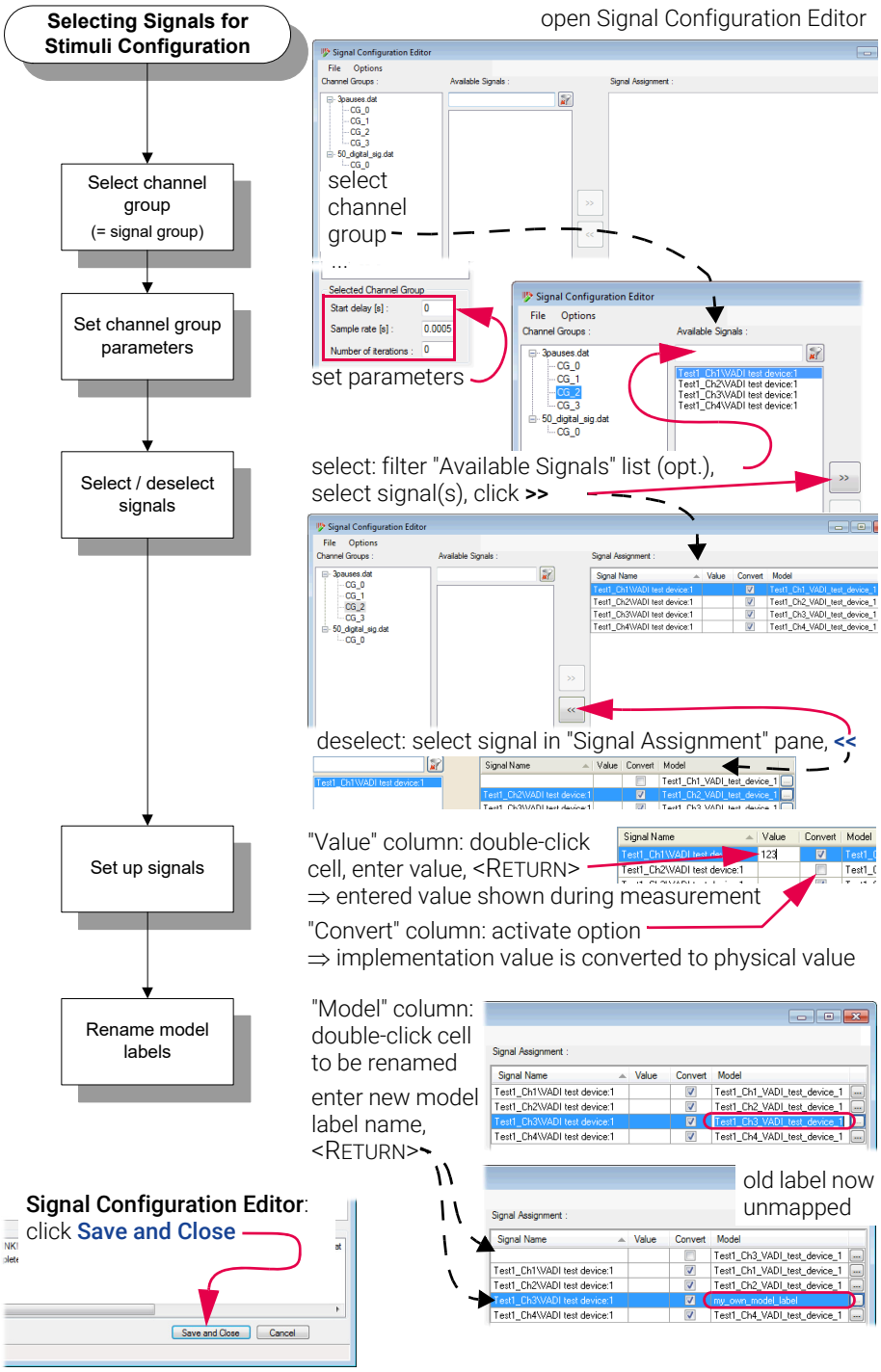



**Fig. 4-18** Signals for stimuli configuration

## 4.5.2 Stimuli Signal Group Receive Block

The Stimuli Signal Group Receive block enables you to create ports by mapping signals defined in existing MDF files to the block. You can route signals via these ports to objects within the control function.

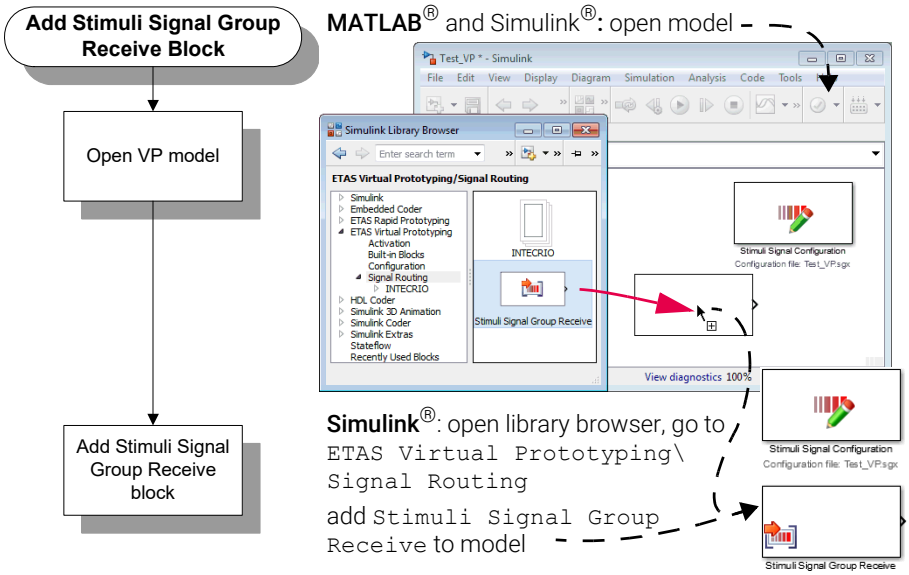



**Fig. 4-19** Adding a Stimuli Signal Group Receive block

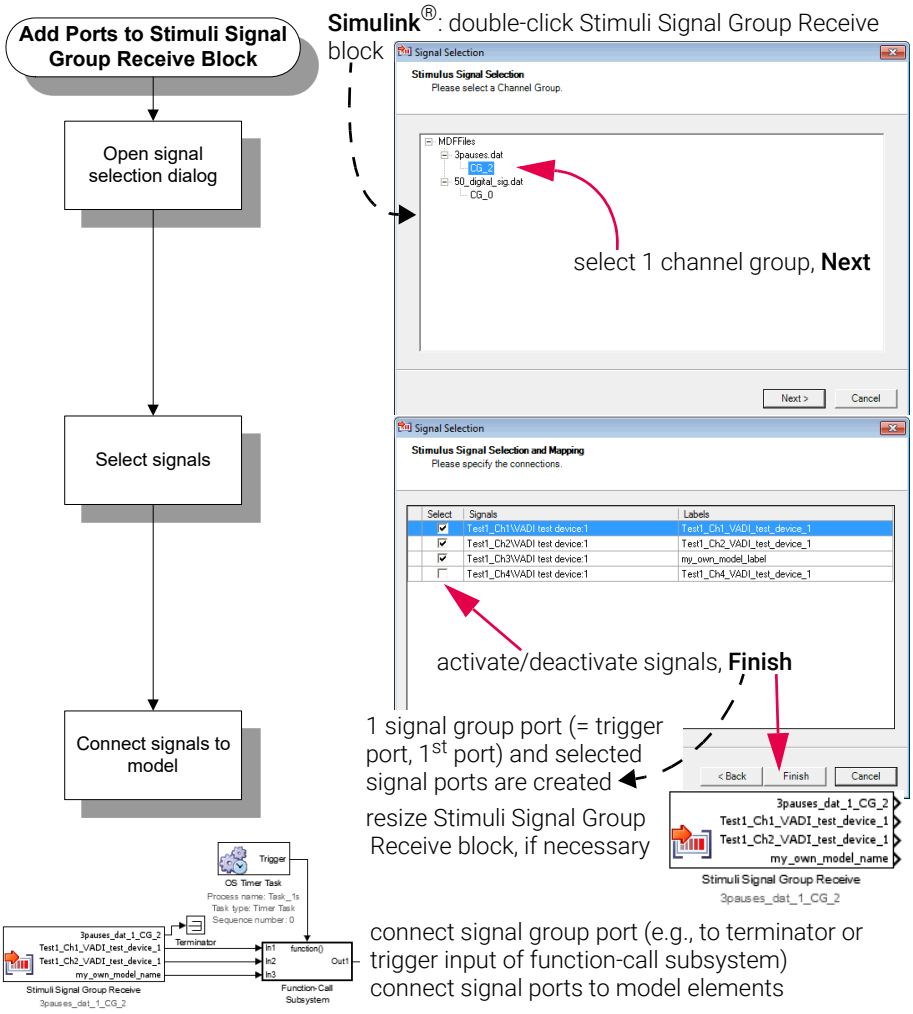



**Fig. 4-20** Adding ports to a Stimuli Signal Group Receive block

Further mapping of signals to ports is done in the Signal Configuration Editor.

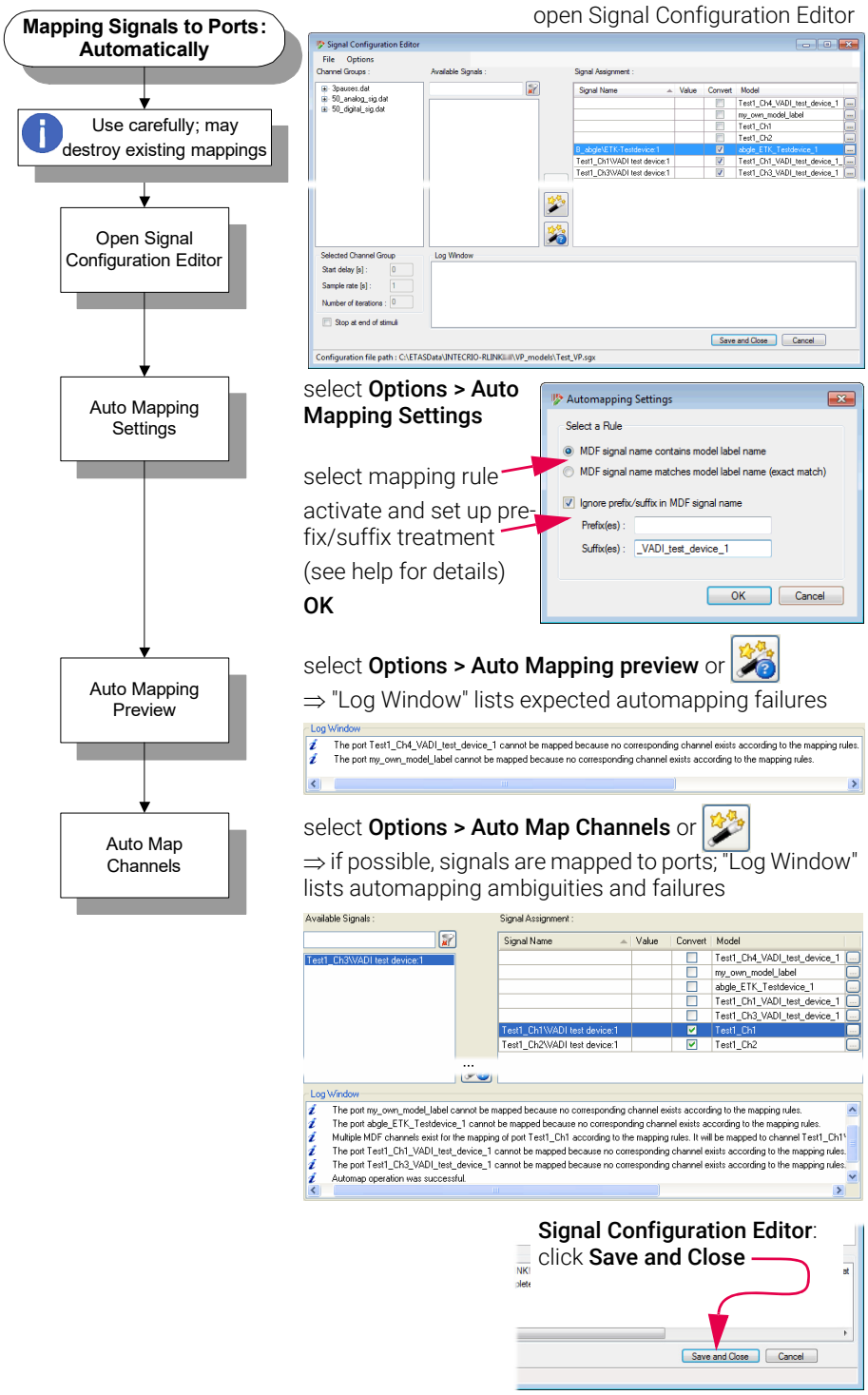



**Fig. 4-21** Mapping signals to ports automatically

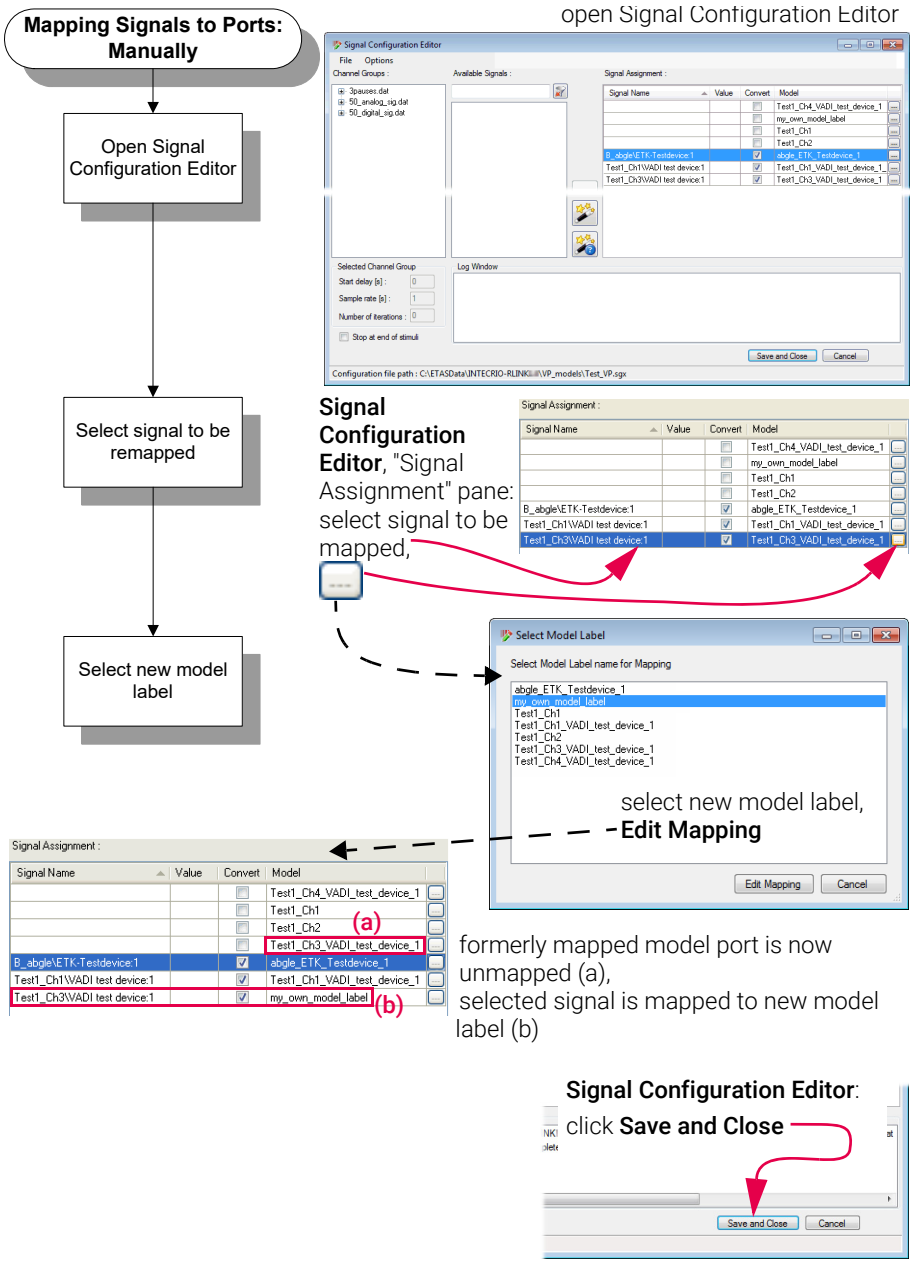



**Fig. 4-22** Mapping signals to ports manually

## 4.6 Exporting the Model and Experimenting

Once your model is complete, you can export it to INCA (chapter 4.6.1) or INTECRIO (chapter 4.6.2), and then perform an experiment. To do so, you need the following software versions, in combination with valid licenses:

- INCA V7.2.17 or higher / INCA-EIP V7.2.17 or higher / ETAS Virtual OS Execution Platform
- INTECRIO V5.0 or higher

## 4.6.1 Export / Experiment – INCA

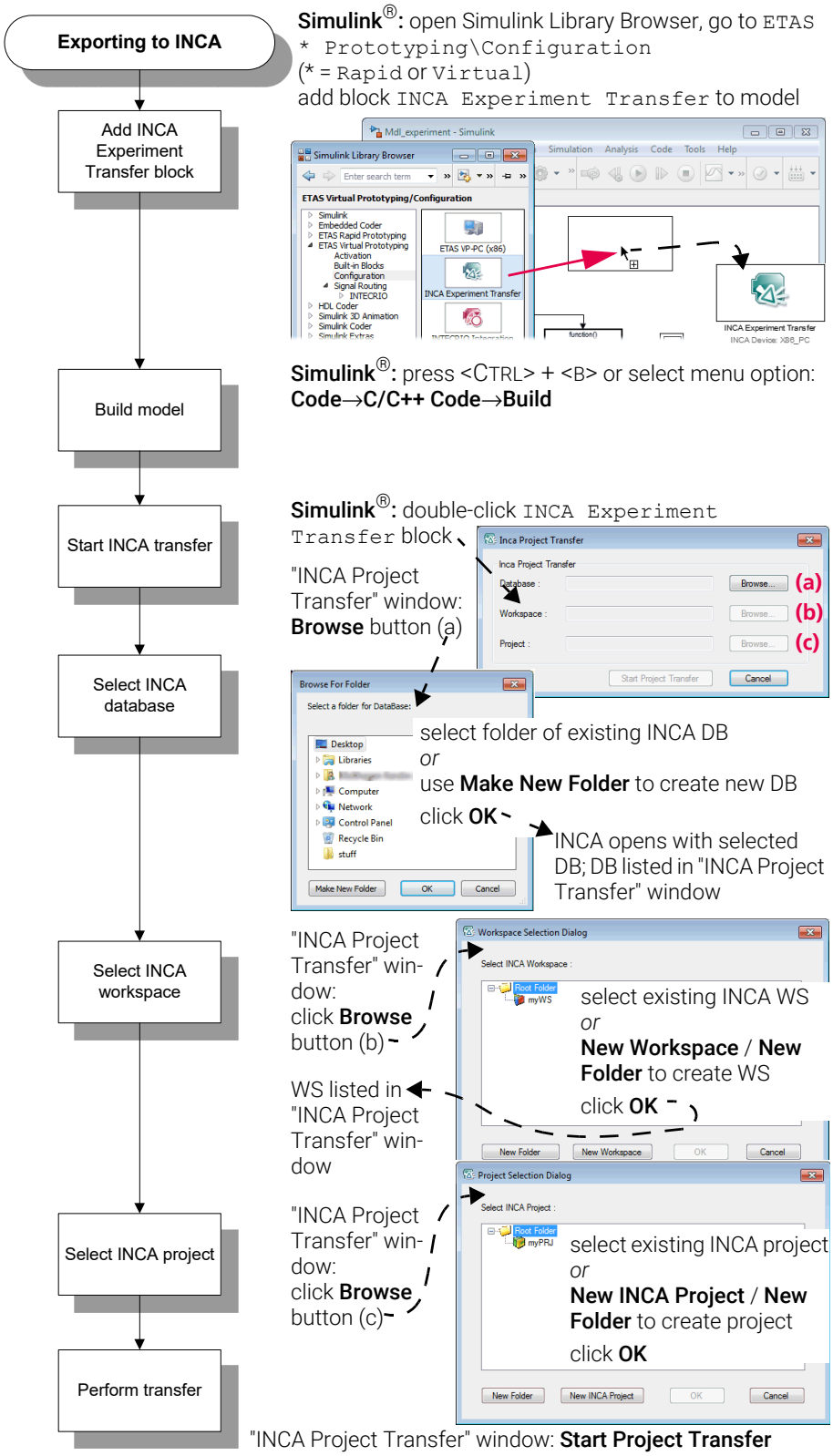



**Fig. 4-23** Exporting the model to INCA

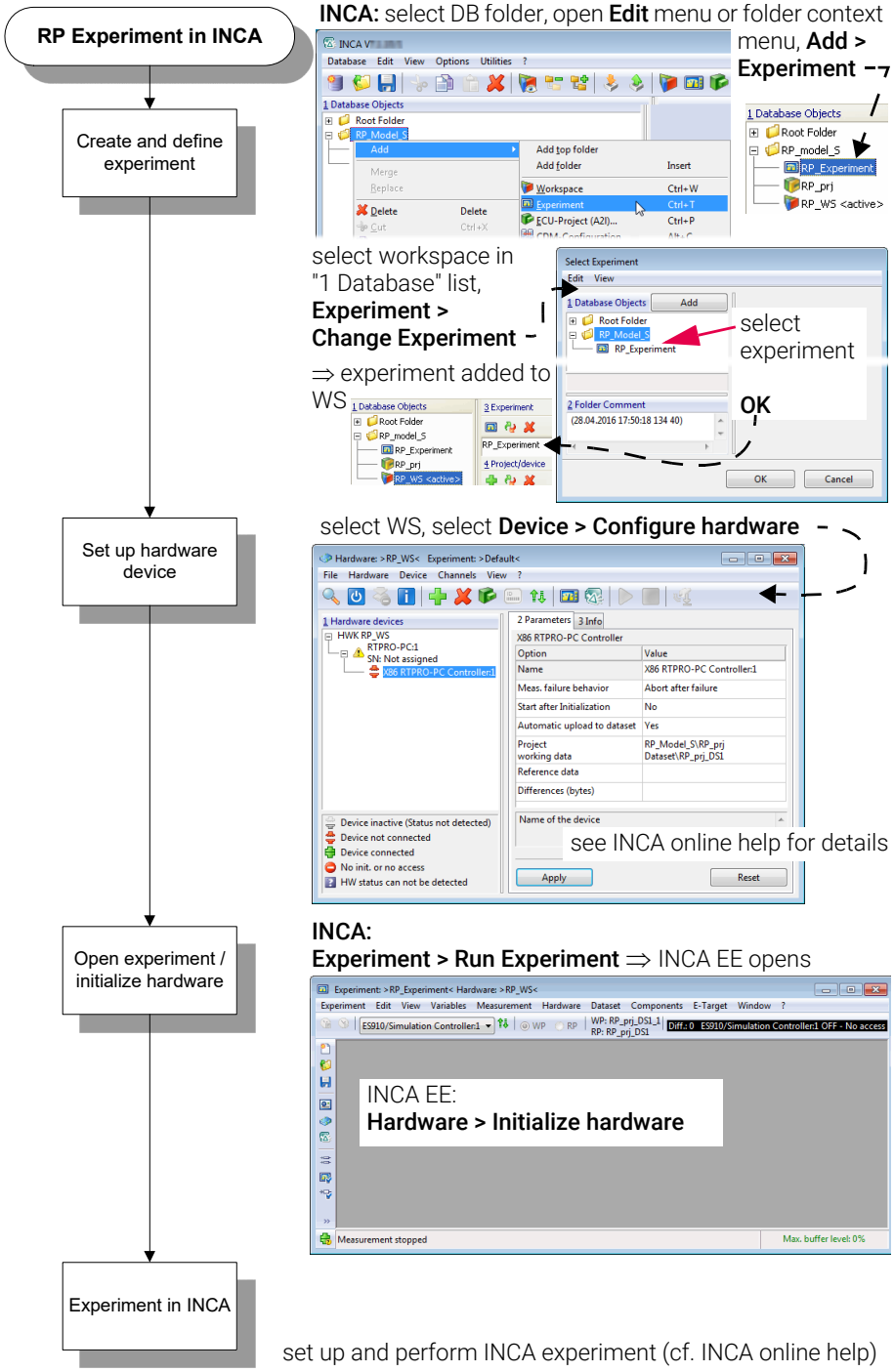



**Fig. 4-24** RP Experiment in INCA

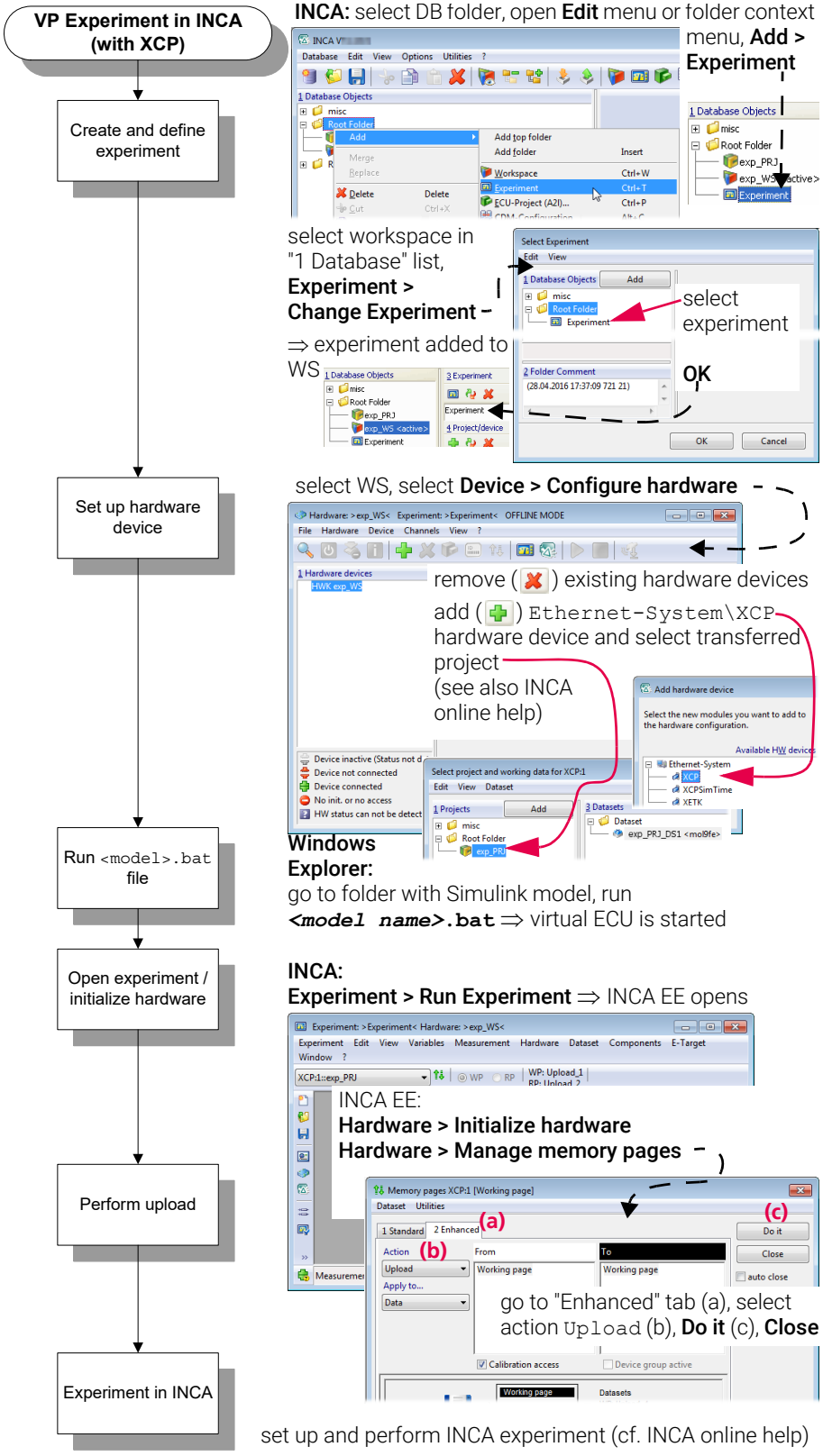



**Fig. 4-25** VP Experiment in INCA (with XCP)

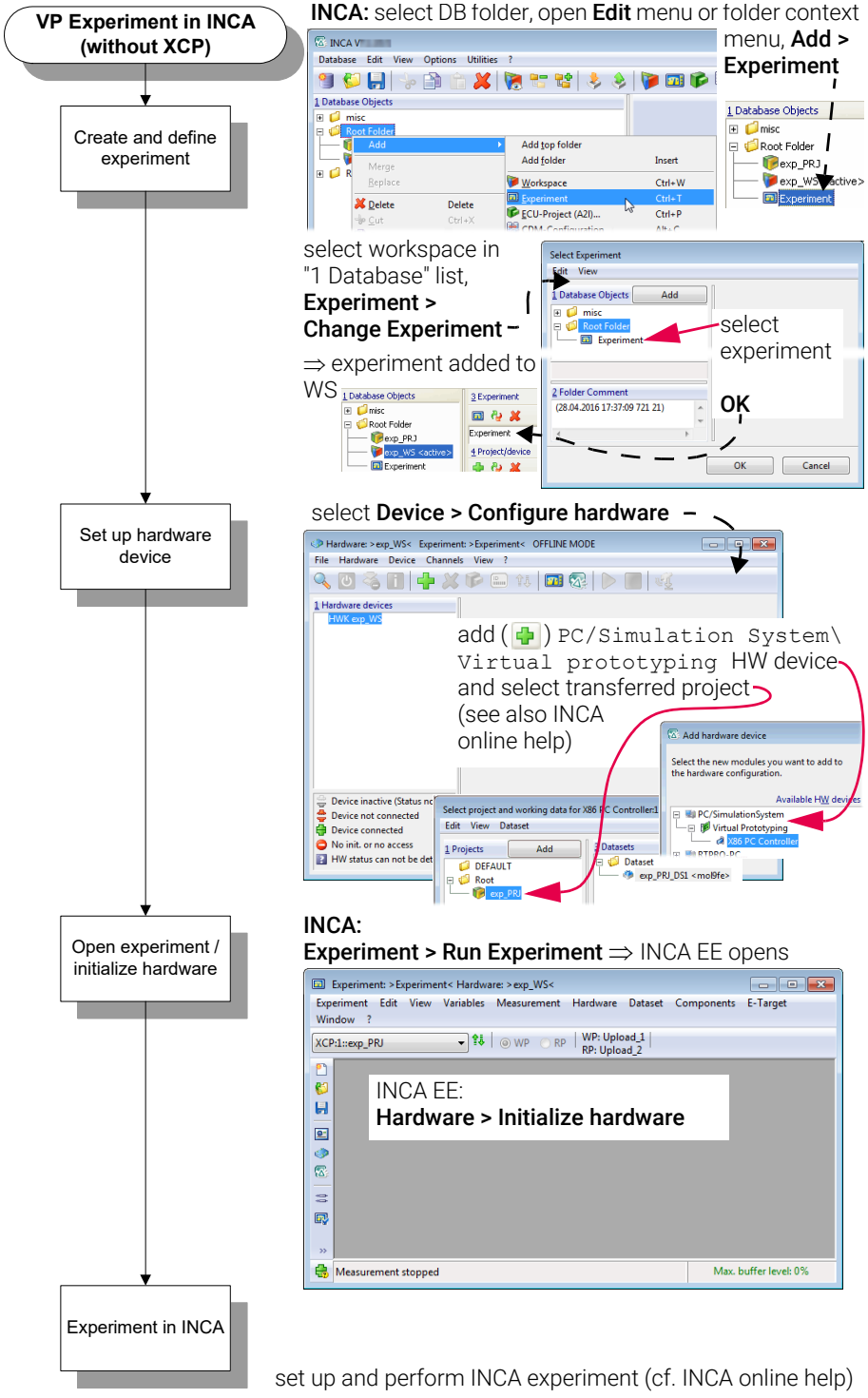



**Fig. 4-26** VP Experiment in INCA / INCA-EIP (without XCP)

## 4.6.2 Export / Experiment – INTECRIO + ETAS Experiment Environment

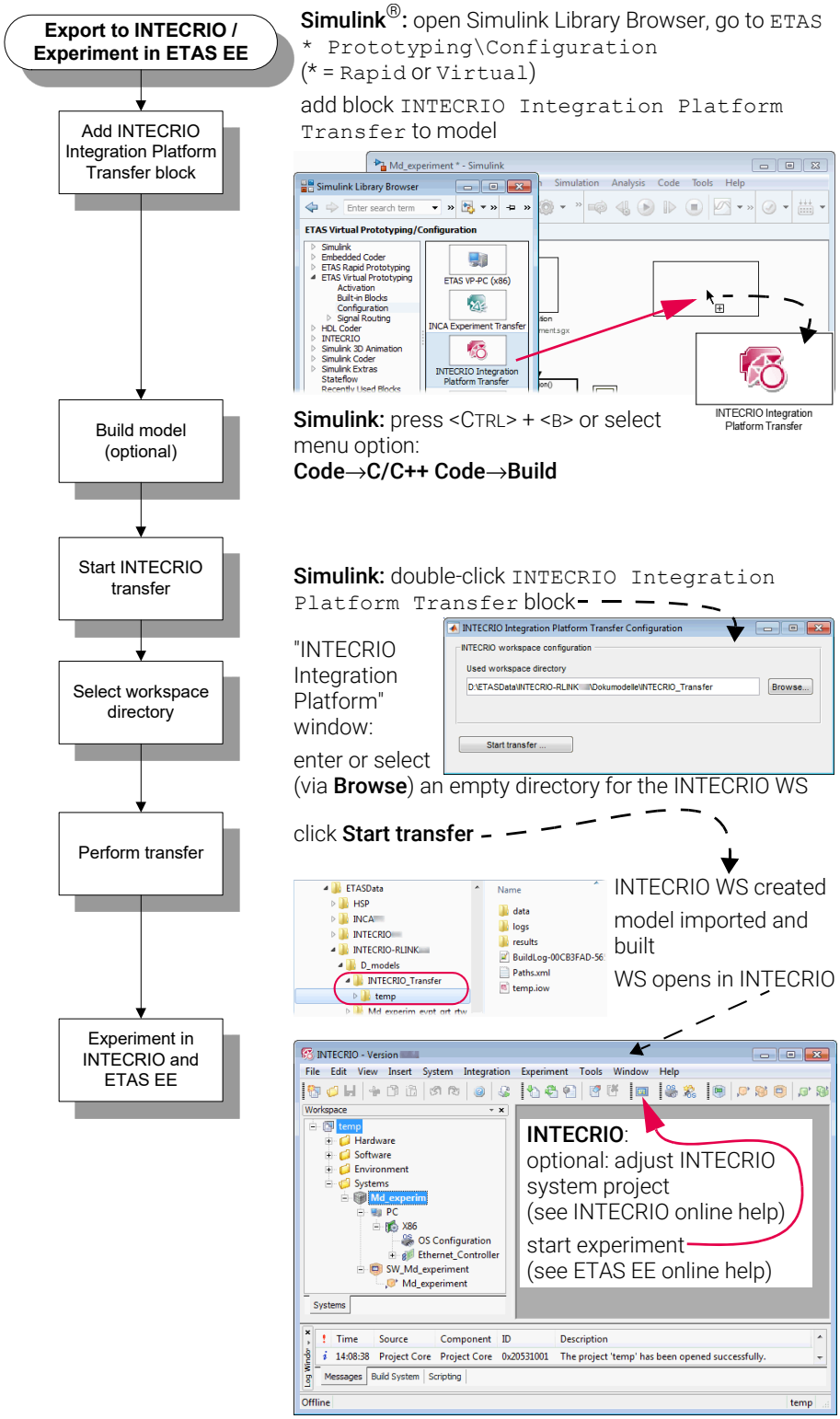



**Fig. 4-27** Exporting the model to INTECRIO, experimenting in ETAS Experiment Environment

# 5 Bypass Concept

**NOTE**

Only Rapid Prototyping targets allow bypass experiments.

## 5.1 ETK Bypass Concept Description

With an ETK equipped ECU, the ECU code must be prepared to set up data structures and communication with the ETK to enable communication between the rapid prototyping system used for the ETK bypass and the ECU itself.

Independent of the ECU implementation of these drivers, some safety issues common to the concept of the bypass must be considered.

## 5.2 Bypass Input

Like for measurement, the Distab13 (and Distab12 for hook-based bypass) mechanism is used to provide ECU variables as inputs for the bypass.

The DISplay TABle for the Distab13 contains a sorted list of addresses of variables in the ETK Flash. 8, 4, 2 and 1 byte values are supported. The addresses are ordered corresponding to the size of the values they point to. The ECU driver parses the table and writes the contents of the addresses to a table of return values in the ETK RAM. This table is ordered in the same manner as the address list: first, all 8 byte values, then the 4 byte values, etc.

With this approach, INCA and INTECRIO-RLINK are given access to values of the internal memory of the microcontroller. Also, collecting the data in a table allows using block modes for transferring the data to the PC.

Fig. 5-1 gives an overview on the data layout for Distab13.

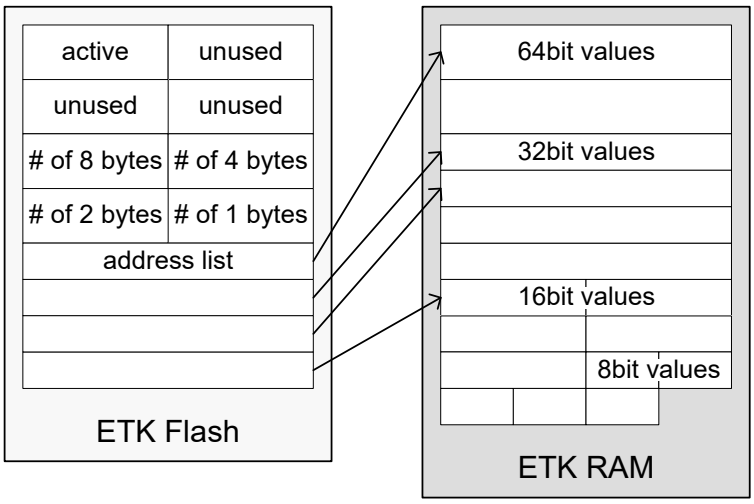



**Fig. 5-1** Distab13 data structure

For each bypass raster that contains hooks for the hook-based bypass, an instance of the Distab is created and a Distab process is called. For the service-based bypass, one instance of the Distab data structure exists for each trigger where a service is provided and configured to read ECU values as input for the bypass calculation.

For the hook-based bypass, the number and name of Distabs implemented in the ECU code, and their size, e.g. the number of bytes per channel, is set by the ECU software setup and documented in the A2L description.

For service-based bypass, all tables are allocated dynamically in a given working memory.

## 5.3 Hook-Based Bypass

### *Classical*

For the classical *hook-based bypass*, the input values of the bypass are gathered with the same Distab13 mechanism as the measurements. After the bypass input data is written to the ETK RAM, the bypass calculation is triggered. In addition, a channel for writing back bypass results to the ETK where they can be retrieved by the ECU is introduced.

For each bypass input channel, an output channel is provided. The size of these channels, as well as their names, also have to be documented in the A2L description. The prototyping tool (INTECRIO-RLINK, INTECRIO, or ASCET-RP) can define the number of variables written back to the ECU, depending on the bypass experiment setup. Each variable written to by the bypass must be prepared in the ECU software by applying a hook to prevent the ECU from writing to this value if the bypass is active. The hook code is specific to the ECU and the variable it is applied to. No service is provided within this sample implementation for this task. The values prepared have to be documented in the A2L file by an `IF_DATA ASAP1B_BYPASS` description.

The following figure describes the hook-based bypass principle. The hook indicates the possibility to toggle between the results of the original function (Fn) and the bypass function (Fn*).

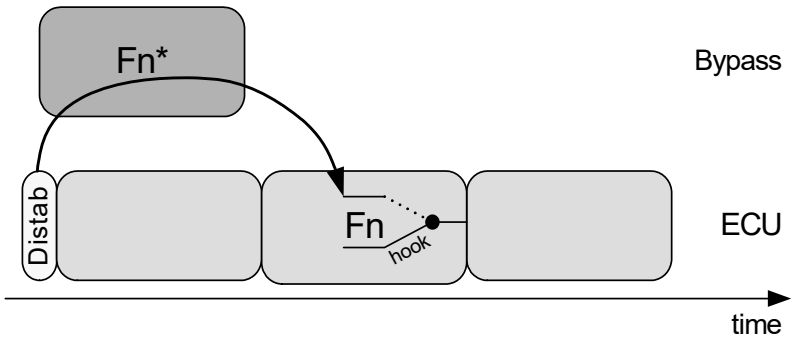



**Fig. 5-2** Hook-Based Bypass: Principle

### *With Distab17*

In connection with Distab17, the concept of service point configuration extends to the hook-based bypass. In this case, the hook-based bypass (HBB) is integrated with the service-based bypass (SBB): For hooked service points, the new signal values calculated in the rapid prototyping system are transferred to the ECU software by dedicated hook codes in the ECU functions instead of generically in a service point write action. Hooked service points are supported as of Distab17.

- In the following illustration, **Fn** denotes the original function that runs on the ECU.

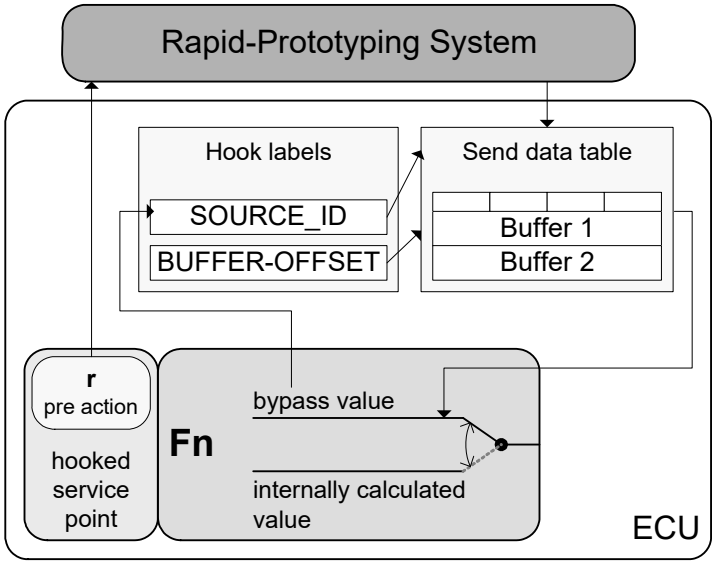



**Fig. 5-3** Bypass with hooked service points

- Prior to the execution of the original function that runs on the ECU, hooked service points can be used to receive data from the ECU to the rapid prototyping system via a read or receive pre-action. The associated hook codes are usually implemented at the end of the original function. The hooks receive their source (i.e. SOURCE_ID) and offset (i.e. BUFFER_OFFSET) information from the associated hook labels.
- During the execution of the original function in the ECU, the rapid prototyping system writes data to a double-buffered send data table that can be accessed by these hooks in the original function. The two buffers are used alternately. Either the resulting bypass value or the value calculated internally by the original function is used.

## 5.4 Service-Based Bypass

**NOTE**

Service-based bypass on an ETK with 8 Mbit/s is not supported.

For the *service-based bypass*, both the input values and the output values of the bypass function are transmitted with the same Distab13 mechanism. After the bypass input data is written to the ETK RAM, the bypass calculation is

triggered. In addition, a channel for writing back bypass results to the ETK where they can be retrieved by the ECU is introduced. Here, each bypassed ECU process uses and calls its own Distab.

This service also contains an inverted Distab mechanism to write back bypass outputs to the ECU. The ECU does not need to apply hooks to the variables written to, since the service simply overwrites the values with the bypass outputs. INTECRIO-RLINK sets up a Distab-like sorted address table with the addresses of the ECU variables to be written to, and writes the corresponding values in a table of the ETK Flash. The part of the ECU service that writes the bypass outputs to the ECU parses the address table and gets the corresponding values from the data table and writes the values to the ECU addresses.

The following figure describes the service-based bypass principle.

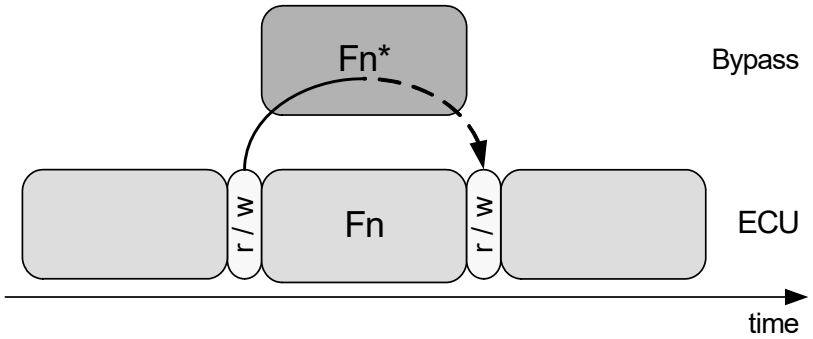



**Fig. 5-4** Service-Based Bypass: Principle (The dashed line indicates that bypass data can be written back at a later time as well.)

INTECRIO-RLINK V5.0 supports several versions of service-based bypass (SBB). Tab. 5-1 lists the supported SBB versions for each target (+: supported, –: not supported), and Tab. 5-2 lists the AML versions available in the SBB versions.

| | SBB V2.0 | SBB V2.1 | SBB V3.* |
|---|---|---|---|
| ES 910.2 / supported ETKs | + | + | – |
| ES 910.3 / supported ETKs | + | + | + |
| ES 8xx / supported ETKs | + | + | + |

**NOTE**

Service-based bypass on an ETK with 8 Mbit/s is not supported.

**Tab. 5-1** Supported SBB versions

| SBB version | supported AML versions | | supported Distab types |
|---|---|---|---|
| V2.0 + V3.0 | ETK AML | 1.2.0 – 1.7.0 | Distab13 - Distab16 |
| | XETK AML | ≥ 1.0.0 | |
| | SBB AML | ≥ 2.0.0 | |
| V2.1 | ETK AML | not supported | Distab17 |
| | XETK AML | ≥ 2.0.0 | |
| | SBB AML | ≥ 3.1.1 | |
| V3.1 | ETK AML | not supported | Distab17 |
| | XETK AML | ≥ 2.0.0 | |
| | SBB AML | ≥ 2.0.0 | |

**Tab. 5-2** SBB versions and AML versions

## 5.5 Bypass Safety Considerations

With calculating a bypass function on a rapid prototyping system and feeding data back into the ECU, the same care needs to be taken for development and use of bypass software as for ECU software. The bypass output may directly or indirectly influence the output channels of the ECU. The same applies, of course, if the rapid prototyping system uses own output channels.

Thus, it is highly recommended that the bypass functions include range checks and validations algorithms for the bypass outputs.

### 5.5.1 Bypass Input Data

To perform proper calculations, the bypass obviously needs consistent and valid input data. The ECU software must ensure this and prevent activation of the bypass if the ECU software detects incorrect or invalid inputs. This also includes ECU states like initialization and afterrun, or error modes the ECU might run in. In other words, activation of and data transfer to the bypass must be covered by the safety mechanisms of the ECU.

### 5.5.2 Bypass Calculation

The ECU software must be aware whether the bypass is active and must provide measures to react on bypass failures, for example missing calculations or unexpected shut off of the bypass system.

Failsafe measures might be using the alternative output values of the bypassed ECU functions, using constant fallback values, or even resetting the ECU, depending on the bypassed ECU function. This is entirely under responsibility of the ECU provider who integrates the bypass drivers.

Some implementations of the ECU bypass drivers, as for the service-based bypass, allow the deactivation of the bypassed ECU function by the bypass user. In this case, the results of the bypassed ECU function obviously cannot be used as fallback. This must be considered when setting up a safety strategy.

### 5.5.3 Bypass Output Data

The ECU provider must guarantee that any value sent back by the bypass system leads to a predictable behavior of the ECU – the bypass output values must undergo the same range check and validation as the values calculated within the bypass.

As said before, the implementation of the ECU drivers must, in any case, ensure that bypass failures can be detected by the ECU and valid and safe fallback values are available at any time.

### 5.5.4 Message Copies

If the ECU software contains message copies, the bypass must be aware of them.

The usual implementations of the *hook-based bypass* are an exception to that rule. Here, the hooks (and thus the messages written to) are known before compilation, so that the hook code can take care of and use message copies if needed—individual code and addresses are used at each hook.

With the *service-based bypass*, users cannot choose variables and decide which message to write to before they set up the bypass experiment in INTECRIO-RLINK. Thus, the services in the ECU are generic code and do not know about specific message copies.

This requires two steps:

A The ECU software provider must provide this message copy information in the A2L file (usually in encrypted and password-protected form).

B If the message copy information is encrypted, the user of the bypass system receives a password from the ECU software provider. He must enter this password to decrypt the information and use it for system configuration.

If one of these two steps is missing (usually a wrong password is entered), the bypass system has no knowledge of message copies and reads from / writes to the original variable address, as it is declared in the `MEASUREMENT` declaration of the A2L file. For receive variables, this yields old data values. For send variables, the data value written by the bypass model may be overwritten by other parts of the ECU software. Both cases may cause bypass malfunction!

Another principal problem can arise with ECU software that contains message copies. The original code to create the message copies for an ECU task was based on the given message usage and generated appropriate code. By writing variable values into the ECU via bypass methods, the data flow changes, and a new or different message copy may become necessary. This can result in wrong variable values in the ECU software even at locations which are not directly related to the bypassed functions. Whether writing to a certain variable at a certain location (e.g. service point) may be dangerous or not, can be answered only by the ECU software supplier. This information cannot be declared in the A2L file.

## 5.6 Service-Based Bypass Specifics

Unlike the hook-based bypass where either the ECU or the bypass writes to the bypassed variable, the service-based bypass has an inherent possibility of data inconsistency, since both the ECU and the bypass write to the same value consecutively. Due to ECU real-time constraints, interrupts cannot be disabled to protect the sequence of writing the results of the ECU function and then writing the variable values of the bypass.

So, if a preemptive task of higher priority interrupts the tasks containing the bypass service, it will see the ECU value instead of the bypass value. The probability of this inconsistency depends on the distance between the two writes.

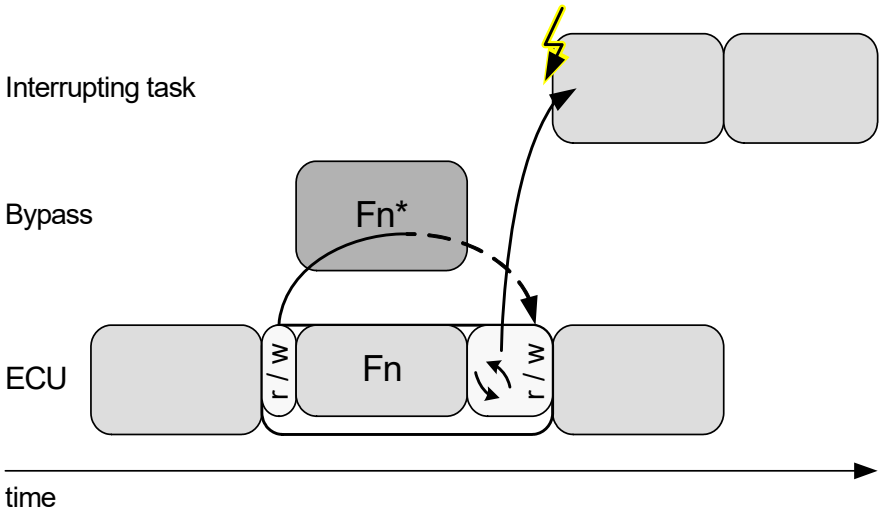



**Fig. 5-5** Possible data inconsistency (the small arrows ( ) indicate a waiting time for bypass results)

A countermeasure to this problem is disabling the ECU function, so that only the bypass is writing to the bypassed ECU values (see below). Be aware that disabling the ECU function implies other safety constraints in case of bypass failures as discussed below.

### 5.6.1 Service Processes for the SBB Implemented as Service Functions

For SBB, the way of ECU implementation is to replace the ECU process to be bypassed by a container process that contains service function calls before and after it calls the original ECU process. This allows to call the ECU process under certain conditions only, e.g. to deactivate it in case of possible data consistency problems. To simulate the timing behavior of the disabled ECU process, a delay time can be configured.

Therefore, the suggested ECU implementation looks like this:

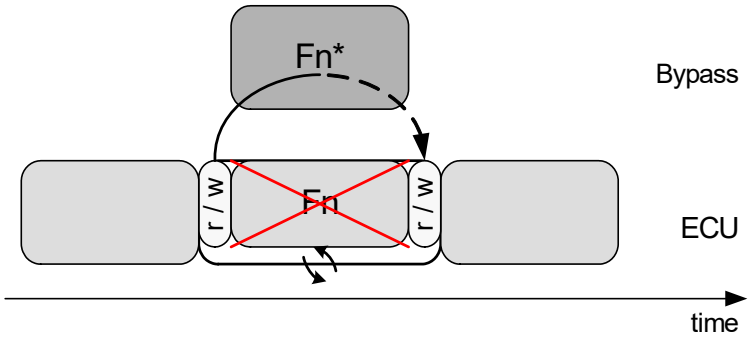



**Fig. 5-6** Suggested SBB implementation

For setting up the bypass experiment in INTECRIO-RLINK and implementing the service point in the ECU software as container process, a service point is defined as

A receiving data from the ECU (pre read action),
B waiting for data to be sent (a time-out is defined),
C sending data to the ECU (pre write action),
D conditionally executing the original ECU process,
E receiving data from the ECU (post read action),
F waiting for data to be sent (a time-out is defined),
G sending data to the ECU (post write action).

Each pre and post action can be freely configured or activated / deactivated.

## 5.6.2 Controlling the ECU Behavior from INTECRIO-RLINK

Upon setting up the INTECRIO-RLINK experiment, an initial setting of the control variables can be done in INTECRIO-RLINK. These values are written to the ETK on experiment initialization.

The ECU function can be controlled by the INTECRIO-RLINK user in several ways (if the ECU drivers also provide this functionality):

- The ECU function can be deactivated in the service point editor of INTECRIO-RLINK ( column, see Fig. 5-7)
- The detection of a bypass error can be defined

- The bypass error behavior of the ECU code can be influenced

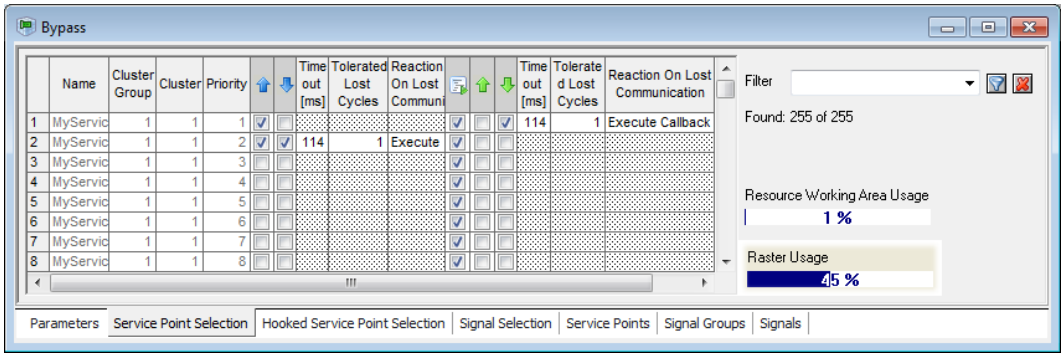



**Fig. 5-7** Controlling ECU function execution from INTECRIO-RLINK (Columns "Cluster Group" and "Cluster" only available for SBB V3.*. The "Raster Usage" display is only available for SBB V2.*.)

## 5.6.3 Model Configuration for Service-Based Bypass V3

This section shows how to configure a Simulink model with service-based bypass V3.

In INTECRIO-RLINK, a service point can consist of at maximum four individual signal groups:

- the pre-action read signal group
- the pre-action write signal group
- the post-action read signal group
- the post-action write signal group.

Each of these signal groups can be represented and scheduled in the model using individual Hardware Signal Group Receive and Send blocks. For details on scheduling, see the online help.

### 5.6.3.1 Restrictions

The following restrictions are valid for the configuration of a Simulink model with service-based bypass V3:

- The currently used RTA-OSEK restricts the number of tasks available for service-based bypass to a maximum of 253 (the total number of available tasks is 256, and 3 system-internal tasks are necessary in any case).

  However, depending on the details of the service point specification in the A2L file, much more than 253 *service points* can be used simultaneously in most cases.
- A service point write action (represented by a `Hardware Signal Group Send` block) can be used only within the subsystem triggered by the associated service point's `Hardware Signal Group Receive` block.

### 5.6.3.2 Classical ECU Function Bypass

In a classical ECU function bypass, the pre read action and the post write action of a service point are activated. Both actions are mapped to the same OS task.

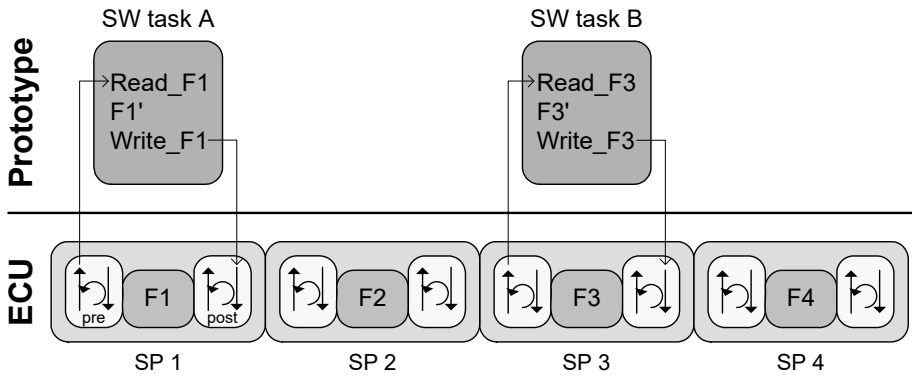
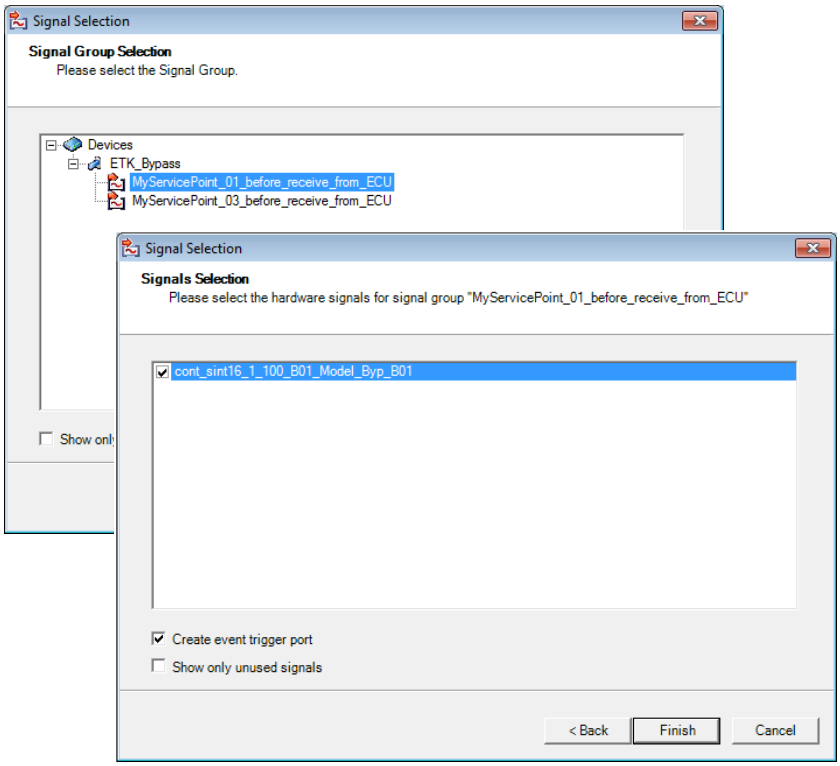



**Fig. 5-8** Classical bypass

To achieve this kind of configuration, the following steps are required:

A For each service-point whose function shall be bypassed, add one `Hardware Signal Group Receive` block for the pre read action and one `Hardware Signal Group Send` block for the post write action.

B Double-click the `Hardware Signal Group Receive` blocks and use the "Signal Selection" window to select the appropriate signal groups and signals, and create the event trigger ports.




C For the `Hardware Signal Group Send` blocks, select the appropriate signal groups and signals.

D Connect the signal blocks with the function-call subsystems that contain the bypass functions.

The `Hardware Signal Group Send` blocks can also be placed inside the respective function-call subsystems.

E When you build the Simulink model, the appropriate OS configuration is generated.

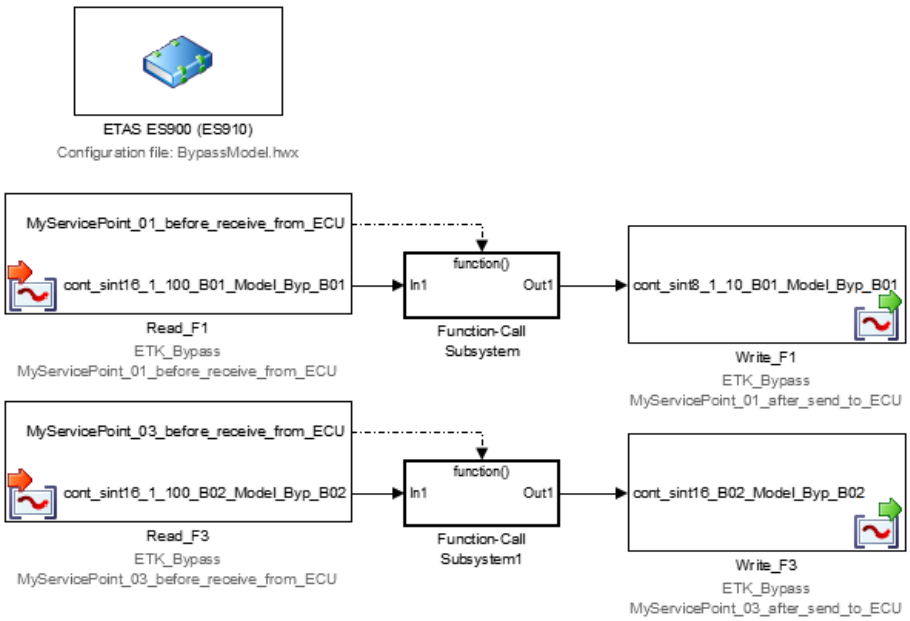



**Fig. 5-9** Simulink configuration for the classical bypass in Fig. 5-8

## 5.6.4 Summary

The implementation and integration of the ECU drivers must take care of the following:

- As the service-based bypass only overwrites ECU values with bypass values without preventing the ECU from writing these values, there is a possibility of data inconsistency which can lead to unpredictable behavior of the ECU.
- INTECRIO-RLINK provides a configuration variable to set the maximal number of tolerated lost cycles, e.g. how many ECU calculation cycles without receiving bypass values are tolerated before this is regarded as an error. But it is up to the provider of the ECU software to make sure missing bypass output values are detected.
- INTECRIO-RLINK provides a configuration variable to set a specific error behavior (if also supported by the ECU implementation). But it is up to the provider of the ECU software to make sure bypass failures or bypass deactivation can be detected by the ECU software! The configuration setting in the INTECRIO-RLINK GUI can then be used to choose between different provided error behaviors in the ECU.
- INTECRIO-RLINK allows disabling the bypassed ECU process. In this case, no ECU values can be used as fallback values for bypass failures! It is up to the provider of the ECU software to make sure sensible data is written to the variables if both the ECU process and the bypass is disabled.

# 6 Troubleshooting General Problems

This chapter gives some information of what you can do when problems arise that are not specific to an individual software or hardware product.

## 6.1 Network Adapter cannot be selected via Network Manager

### *Cause: APIPA is disabled*

The alternative mechanism for IP addressing (APIPA) is usually enabled on all Windows systems. Network security policies, however, may request the APIPA mechanism to be disabled. In this case, you cannot use a network adapter which is configured for DHCP to access ETAS hardware. The ETAS Network Manager displays a warning message.

The APIPA mechanism can be enabled by editing the Windows registry. This is permitted only to users who have administrator privileges. It should be done only in coordination with your network administrator.

To enable the APIPA mechanism

1. Open the Registry Editor:
   i. Press <WINDOWS LOGO> + <R>.
   ii. Enter `regedit` and click **OK**.
   The registry editor is displayed.
2. Open the folder `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpic\Parameters\`.
3. Select **Edit > Find** to search for the key `IPAutoconfigurationEnabled`.

If you cannot find any instances of the registry key mentioned, the APIPA mechanism has not been disabled on your system, i.e. there is no need to enable it. Otherwise proceed with the following steps.

4. Set the value of the key `IPAutoconfigurationEnabled` to `1` to enable the APIPA mechanism.
   You may find several instances of this key in the Windows registry which either apply to the TCP/IP service in general or to a specific network adapter. You only need to change the value for the corresponding network adapter.
5. Close the registry editor.
6. Restart your workstation in order to make your changes take effect.

## 6.2 Search for Ethernet Hardware fails

There is a number of different causes which might lead to connection problems, most of them being based on inappropriate Windows or hardware settings. Usually, these can easily be modified, once they have been identified.

The following list of causes shall help you in finding the root cause of the problem and fixing it.

### *Cause: The versions of the Hardware and the ETAS Software are not compatible*

If you are using ETAS hardware with ETAS software, you can use the ETAS HSP Update Tool to check the firmware version of your hardware:

- Make sure you use the ETAS HSP Update Tool with the latest HSP (Hardware Service Pack) version.
- Also use the HSP Update Tool to check whether the hardware is compatible with the software used.
- Make sure any additional drivers for that hardware are installed correctly.

You can get the required HSP from the ETAS internet pages at www.etas.com.

If you still cannot find the hardware using the HSP Update Tool, check whether the hardware offers a Web interface and whether you can find using this interface. Otherwise check whether one of the following causes and solutions might apply.

### *Cause: Personal Firewall blocks Communication*

Personal firewalls may interfere with access to ETAS Ethernet hardware. The automatic search for hardware typically cannot find any Ethernet hardware at all, although the configuration parameters are correct.

Certain actions in ETAS products may lead to some trouble if the firewall is not properly parameterized, e.g. upon opening an RP experiment in the ETAS experiment environment or searching for hardware from within INCA or HSP.

If a firewall is blocking communication to ETAS hardware, you must either disable the firewall software while working with ETAS software, or the firewall must be configured to give the following permissions.

Permissions given through the firewall block ETAS hardware:

- Outgoing limited IP broadcasts via UDP (destination address 255.255.255.255) for destination ports 17099 or 18001
- Incoming limited IP broadcasts via UDP (destination IP 255.255.255.255, originating from source IP 0.0.0.0) for destination port 18001
- Directed IP broadcasts via UDP to the network configured for the ETAS application, destination ports 17099 or 18001
- Outgoing IP unicasts via UDP to any IP in network configured for the ETAS application, destination ports 17099 through 18020
- Incoming IP unicasts via UDP originating from any IP in the network configured for the ETAS application, source ports 17099 through 18020, destination ports 17099 through 18020

- Outgoing TCP/IP connections to the network configured for the ETAS application, destination ports 18001 through 18020

  **NOTE**

  The ports that have to be used in concrete use cases depend on the hardware you use. For more precise information on the port numbers that can be used please refer to your hardware documentation.

Permissions given through the firewall block XCP on Ethernet:

- Outgoing IP multicasts for XCP Slave Detection via UDP to any IP in network, destination IP 239.255.0.0, port 5556.
- Incoming IP multicasts for XCP Slave Detection via UDP from any IP in network, destination IP 239.255.37.45, port 3745.

Contact your IT responsible to clarify whether the required permissions are, or can be, given by the firewall.

***NOTICE***

**Insecure system due to changes to your firewall**

Always consult your IT responsible and/or check the IT security policies of your company before changing your firewall configuration and reconnecting the computer to the network.

### *Cause: Client Software for Remote Access blocks Communication*

PCs or notebooks which are used outside the ETAS hardware network sometimes use a client software for remote access which might block communication to the ETAS hardware. This can have the following causes:

- A firewall which is blocking Ethernet messages is being used (see “Cause: Personal Firewall blocks Communication” on page 66)
- By mistake, the VPN client software used for tunneling filters messages. As an example, Cisco VPN clients with versions before V4.0.x in some cases erroneously filtered certain UDP broadcasts.

  If this might be the case, please update the software of your VPN client.

### *Cause: ETAS Hardware hangs*

Occasionally the ETAS hardware might hang. In this case switch the hardware off, then switch it on again to re-initialize it.

### *Cause: ETAS Hardware went into Sleep Mode*

In order to save power, some ETAS devices will go to sleep mode if they do not see that they are connected to another device/computer.

To solve that, connect your Ethernet cable from your computer to the "HOST"/ "Sync In" port on the device. After the device turns on, connect to the device using the web interface and change the settings so that the device stays always on. Consult the device's manual for details on how to do that.

### *Cause: Network Adapter temporarily has no IP Address*

Whenever you switch from a DHCP company LAN to the ETAS hardware network, it takes at least 60 seconds until ETAS hardware can be found. This is caused by the operating system's switching from the DHCP protocol to APIPA, which is being used by the ETAS hardware.

### *Cause: ETAS Hardware had been connected to another Logical Network*

If you use more than one PC or notebook for accessing the same ETAS hardware, the network adapters used must be configured to use the same logical network. If this is not possible, it is necessary to switch the ETAS hardware off and on again between different sessions (repowering).

### *Cause: Device driver for network card not in operation*

It is possible that the device driver of a network card is not running. In this case you will have to deactivate and then reactivate the network card.

To deactivate and reactivate the network card

1. To deactivate the network card, open the Control Panel.
2. Go to the "Network and Sharing Center", then click the "Change adapter settings" link.
3. In the "Network Connections" window, right-click the used network adapter and select **Disable** in the context menu.
4. In order to reactivate the network adapter right-click it again and select **Enable**.

### *Cause: Laptop power management deactivates the network card*

The power management of a laptop computer can deactivate the network card. Therefore you should turn off power monitoring on the laptop.

To switch off power monitoring on the laptop

1. From the Windows Start Menu, select **Control Panel > Hardware and Sound > Device Manager**.
2. In the Device Manager, open the tree structure of the **Network Adapters** entry.
3. Right-click the used network adapter and select **Properties** in the context menu.
4. Select the **Power Management** tab and deactivate the **Allow the computer to turn off this device to save power** option.
5. Select the **Advanced** tab. If the property **Autosense** is included, deactivate it also.
6. Click **OK** to apply the settings.

### *Cause: Automatic disruption of network connection*

It is possible after a certain period of time without data traffic that the network card automatically interrupts the Ethernet connection. This can be prevented by setting the registry key `autodisconnect`.

To set the registry key **autodisconnect**

1. Open the Registry Editor.
2. Select under `HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\lanmanserver\parameters` the Registry Key `autodisconnect` and change its value to `0xffffffff`.

# 7 Contact Information

## ETAS Headquarters

ETAS GmbH

| | | |
|---|---|---|
| Borsigstraße 24 | Phone: | +49 711 3423-0 |
| 70469 Stuttgart | Fax: | +49 711 3423-2106 |
| Germany | Internet: | www.etas.com |


## ETAS Subsidiaries and Technical Support

For details of your local sales office as well as your local technical support team and product hotlines, take a look at the ETAS website:

| | | |
|---|---|---|
| ETAS subsidiaries | Internet: | www.etas.com/en/contact.php |
| ETAS technical support | Internet: | www.etas.com/en/hotlines.php |

# 8 Glossary

This glossary contains explanations of the technical terms and abbreviations used in the INTECRIO-RLINK documentation. While some terms are also used in a more general sense, this glossary specifically addresses the meaning of those terms as they are applied to rapid and virtual prototyping with INTECRIO-RLINK.

The terms are listed in alphabetical order.

**ASAM-MCD**

Working Group for Standardizing Automation and Measuring Systems, including the Working Groups Measuring, Calibrating, and Diagnostics (German: Arbeitskreis zur **S**tandardisierung von **A**utomations- und **M**esssystemen, mit den Arbeitsgruppen **M**essen, **C**alibrieren und **D**iagnose)

**ASAM-MCD-2MC**

A file format used to describe the calibration variables and measured signals contained in the control unit software, and additional specific information designed to parameterize the experiment interface. ASAM-MCD-2MC is used to import the information required for this into an experiment (A2L file).

INTECRIO-RLINK V5.0 supports the ETK AML versions 1.1 (only hook-based bypass) or 1.2 – 1.7 (hook-based and service-based bypass), XETK AML versions up to 2.5, ASAP1B_Bypass AML V1.0 and higher, and SBB AML versions 2.0, 3.0. and 3.1.

For further information, refer to https://www.asam.net.

**BR_XETK**

Emulator test probe (ETK) with Automotive Ethernet interface.

Requires an ES800 hardware system with ES882.

**Bypass experiment**

In a bypass experiment, parts of an electronic control unit program are executed on an experimental target (ES900 or ES800). This requires a special hook in the code.

INTECRIO-RLINK V5.0 supports several types of bypass experiments: XCP bypass on CAN or UDP, as well as hook-based and service-based ETK/XETK/FETK bypass.

**CAN**

**C**ontroller **a**rea **n**etwork

**CANdb**

**CAN d**ata**b**ase; CAN description file created with the CANdb data management program made by the company Vector Informatik.

INTECRIO-RLINK V5.0 supports the CANdb versions V2.3 and higher.

**CPU**

**C**entral **p**rocessing **u**nit

**Distab**

Data exchange method, used in ETK bypass experiments for data exchange between experimental target and ECU.

INTECRIO-RLINK V5.0 supports hook-based bypass with Distab 12 and higher (XETK AML: Distab 13), as well as service-based bypass with Distab 13.

In combination with ES830 or ES910.3 (or later ES910 models once they are available), INTECRIO-RLINK V5.0 supports also bypass with Distab 17.

**ECU**

**E**lectronic **c**ontrol **u**nit; a small embedded computer system that consists of a CPU and the associated periphery, where everything is usually located in the same housing.

**Embedded Coder®**

An add-on for the Simulink® Coder™; extends the capabilities provided by the Simulink Coder to support specification, integration, deployment, and testing of production applications on embedded targets.

**ERPT_ERT**

ETAS rapid prototyping real-time target for embedded coder (Embedded Coder® real-time target)

**ERPT_GRT**

ETAS rapid prototyping real-time target (Simulink® Coder™ real-time target)

**ES4xx**

A series of micro measurement modules (A/D, thermo, Lambda, counter and frequency modules) designed for installation in the immediate proximity of sensors or signals being measured.

**ES63x**

A series of single and dual-channel lambda modules that measure the oxygen content in the exhaust gas with high accuracy.

**ES800 system**

A series of hardware modules that provide a scalable system for validation, application and prototyping powerful electronic control units.

**ES830**

Stackable high-performance rapid-prototyping module of the ES8xx product family, intended for challenging rapid-prototyping applications.

**ES882**

The ES882 ECU and Bus Interface Module supports several interfaces, e.g. CAN and XETK/BR_XETK.

**ES886**

The ES886 ECU and Bus Interface Module supports several interfaces, e.g. CAN and XETK/BR_XETK.

**ES88x**

short for "ES882 and/or ES886"

**ES891**

The ES891 ECU and Bus Interface Module supports several interfaces, e.g. CAN and FETK.

**ES892**

The ES892 ECU and Bus Interface Module supports several interfaces, e.g. CAN and FETK.

**ES89x**

short for "ES891 and/or ES892"

**ES900 system**

An ETAS rapid prototyping system that consists of one ES910.2 or ES910.3 rapid prototyping module and—optional—additional ES920, ES921, ES922, ES4xx, ES63x or ES93x hardware modules.

**ES930**

A compact, rugged, versatile and powerful module with several input and output channels for rapid prototyping, testing and calibration.

**ETK**

emulator test probe (German: **E**mulator-**T**ast**k**opf)

**Event**

An event is an (external) trigger that initiates an action of the operating system, such as a task.

**EVPT_ERT**

ETAS virtual prototyping realtime target for embedded coder Embedded Coder real-time target)

**EVPT_GRT**

ETAS virtual prototyping realtime target (Simulink Coder real-time target)

**FETK**

Fast ECU interface (emulator test probe or ETK) for the ETAS ES89x ECU and Bus Interface Modules)

**FIBEX**

**Fi**eld **B**us **Ex**change – an exchange format based on an XML schema which is used for descriptions of the complete in-vehicle communication network. FIBEX is defined for various network types (CAN, LIN, MOST, FlexRay) and contains information about bus architecture, signals, properties of network nodes, etc.

The FIBEX file format is standardized by ASAM (Association for Standardization of Automation and Measuring Systems).

INTECRIO-RLINK V5.0 supports the FIBEX baseline versions FIBEX V2.0.x and V3.1.0; the latter with some restrictions (see online help for details).

For further information, refer to https://www.asam.net.

**FlexRay**

FlexRay is a scalable and fault tolerant communication system for high-speed and deterministic data exchange. FlexRay's time-division multiplexing facilitates the design of modular or safety-related distributed systems. Its high bandwidth of 10 MBaud on two channels helps to cope with the high network load caused by the increasing amount of innovative electronic systems in modern vehicles.

The communication system's specifications are released by the FlexRay consortium which is widely supported by vehicle manufacturers and suppliers worldwide.

**HBB**

**h**ook-**b**ased **b**ypass

**HC**

**h**ardware **c**onfigurator

**HW**

**H**ard**w**are

**INCA**

ETAS measuring, calibration and diagnostics system (**In**tegrated **C**alibration and **A**cquisition Systems)

INTECRIO-RLINK V5.0 requires INCA V7.2.17 or higher.

**INCA-EIP**

INCA add-on; allows access to the rapid prototyping (ES910, ES830) and virtual prototyping (VP-PC) targets for INCA.

INTECRIO-RLINK V5.0 requires INCA-EIP V7.2.17 or higher.

**INTECRIO Integrated Prototyping Environment**

The INTECRIO Integrated Prototyping Environment offers an integration platform that is able to combine, i.e. integrate, control algorithms whose parts have been created with different function modeling tools. It allows for creating and configuring hardware systems and the connections of these hardware systems with the control algorithms.

INTECRIO-RLINK V5.0 and INTECRIO V5.0 share the same technical basis. INTECRIO-RLINK provides an automated transfer of their models to workspaces for the INTECRIO Integrated Prototyping Environment.

**INTECRIO-RLINK**

INTECRIO-RLINK is a tool that allows users to configure a rapid-prototyping or virtual-prototyping experiment from within MATLAB and Simulink and use it in experiments with INCA/INCA-EIP.

**LDF**

**L**IN **d**escription **f**ile – a configuration file for a LIN controller.

INTECRIO-RLINK V5.0.2 supports the LDF versions 1.3, 2.0, 2.1, and 2.2.

**LIN**

**L**ocal **I**nterconnect **N**etwork; a serial network protocol used for communication between components in vehicles.

LIN is used where the bandwidth and versatility of CAN are not needed. Typical application examples are the networking within the door or the seat of a motor vehicle.

**LSB and lsb**

**L**east **S**ignificant **B**yte (capital letters) or **b**it (small letters)

**MATLAB®**

High-performance language for technical calculations; contains mathematical calculations, visualization and programming in one environment.

**MATLAB® Coder™**

Code generator for MATLAB code.

**MSB and msb**

**M**ost **S**ignificant **B**yte (capital letters) or **b**it (small letters)

**NVRAM**

**N**on-**v**olatile **RAM**

**OS**

**O**perating **s**ystem

**OSEK**

Working group for Open Systems for Electronics in Motor Vehicles (German: **O**ffene **S**ysteme für die **E**lektronik im **K**raftfahrzeug)

**PDU**

**P**rotocol **d**ata **u**nit; a data unit that contains payload and control information which is passed between the layers in a protocol stack.

In INTECRIO-RLINK V5.0, a FlexRay PDU corresponds to a signal group.

**Process**

A process is a simultaneously executable functionality that is activated by the operating system. Processes are specified in modules and do not feature any arguments/inputs or result values/outputs.

**Processor**

see CPU

**Prototype**

Completely executable file for an experimental target system. Such a prototype shows the software functions in practical use – entirely with different goal directions and in a different appropriation.

**Rapid prototyping**

The execution of a software on an experimental target, i.e. a computer with an interface to the vehicle.

**RTA-OSEK**

ETAS real-time operating system; implements the AUTOSAR-OS V1.0 (SC-1) and OSEK/VDX OS V2.2.3 standards and is fully MISRA compliant.

**RTA-OS**

ETAS real-time operating system; implements the AUTOSAR R3.0 OS and OSEK/VDX OS V2.2.3 standards and is fully MISRA compliant.

**RTA-RTE**

AUTOSAR runtime environment by ETAS

**RTIO**

**R**eal-**t**ime **I**nput-**O**utput

**RTOS**

**R**eal-**t**ime **o**perating **s**ystem

**SBB**

**s**ervice-**b**ased **b**ypass

**SCOOP**

**S**ource **Co**de, **O**bjects, and **P**hysics

**SCOOP-IX**

SCOOP **I**nterface E**x**change language.

INTECRIO-RLINK V5.0 supports the SCOOP-IX versions V1.0, V1.1, V1.2, V1.4, and V1.5.

**Service point**

A service point is an encapsulation of a process in the ECU software. It provides data transfer actions to and from the target system; these actions can be enabled and configured by the user.

**Service point cluster**

A group of service points that are executed in the ECU with the same priority (service points located in the same ECU task).

**Service point cluster group**

A group of service point clusters. The group contains all service points of all tasks that can potentially be invoked at the same time in the ECU.

**Simulink®**

Tool for modeling, simulation and analysis of dynamic systems.

INTECRIO-RLINK adds the possibility to configure rapid and virtual prototyping experiments and run them on ETAS hardware using INCA and INCA-EIP.

**Simulink® Coder™**

Code generator for Simulink and Stateflow models. Requires the MATLAB® Coder™.

**Stateflow®**

Tool for modeling and simulation of complex event-controlled systems. It is seamlessly integrated in MATLAB and Simulink.

**SP**

**s**ervice **p**oint

**SW**

**s**oft**w**are

**Task**

A task is an ordered collection of processes that can be activated by the operating system. Attributes of a task are its application modes, activation trigger, priority and modes of its scheduling. Upon activation, the processes of the task are executed in the specified order.

**UDP**

**U**ser **d**atagram **p**rotocol

**Virtual prototyping**

Function developers create virtual prototypes of electronic vehicle functions and test them on the PC.

**XCP**

Universal measurement and **c**alibration **p**rotocol; the **x** generalizes the various transportation layers that can be used. The long name is ASAM MCD-1 XCP.

INTECRIO-RLINK V5.0 supports XCP version V1.0 and all subsequent versions which are compatible with V1.0. In addition, the `XCPplus` keyword from V1.1 and higher is supported.

**XETK**

emulator test probe (ETK) with Ethernet interface

**XML**

E**x**tensible **M**arkup **L**anguage

# Figures

# Index

## L



## M



## O



## P



## R



## S



## T



## U



## V