OneEye_DAS_Shell_1 for KIT_AURIX_TC275_LK

Shell via DAS interface using OneEye

AURIX™ TC2xx Microcontroller Training
V1.0.0

Scope of work

A Shell is used to parse a command line and call the corresponding command execution. A OneEye pipe is used to interface OneEye with the Shell through the DAS interface.

After configuring the OneEye DAS interface, the Shell from iLLDs is used to interpret and manage commands like "info" or "help". The example creates a OneEye pipe and the corresponding DPipe that is used to interface the shell.

Introduction

- **OneEye** is a GUI that enables the creation of interactive Graphical User Interface. Graphical elements can be drag from a toolbox and drop onto the GUI. The behavior of the created GUI can be customized. Different communication interfaces like UART, Ethernet, CAN, DAS can be used to interact with the embedded system

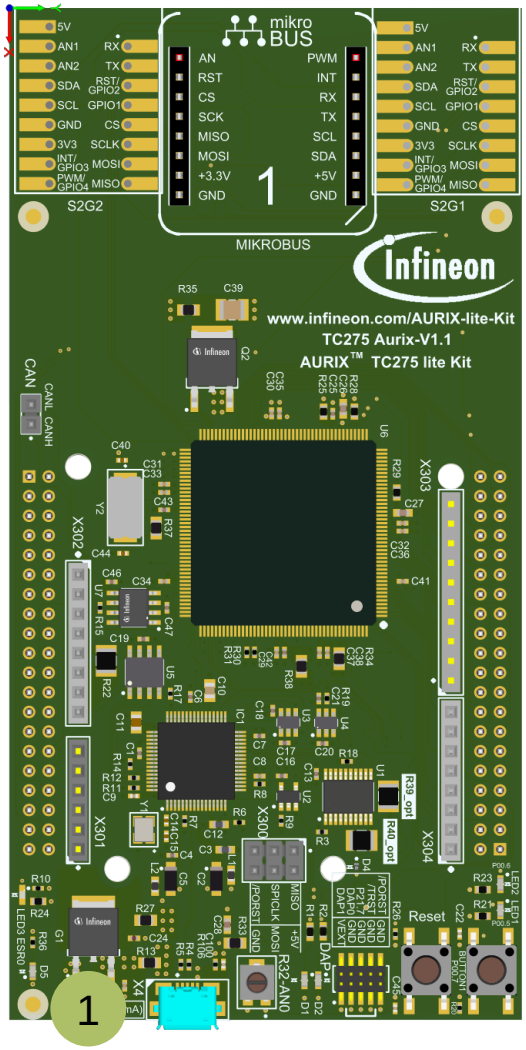
- The **DAS** (Device Access Server) can be used in line with Infineon Microcontroller Starter Kits, Application Kits and DAP MiniWiggler to access the micro controller resources

- **Recommendation**: It is recommended to go through some of the **basic tutorials** listed in the help embedded in OneEye (Menu: Help -> OneEye help). This enables a quicker ramp-up in the OneEye concept and ensure a nice journey with OneEye

Hardware setup

This code example has been developed for the board KIT_AURIX_TC275_LITE.

The board should be connected to the PC through the USB port 1



Implementation - AURIX

Configuring the OneEye pipe

A OneEye pipe (***Ifx_OneEyeDasPipe***) is a special object that is recognized by OneEye and enables streaming of data between OneEye and the microcontroller. The OneEye pipe is initialized with ***Ifx_OneEyeDasPipe_init()***. The ***size*** parameter passed for the configuration corresponds to the buffer allocated for the data transmission, in this example 2 buffers of 512 bytes are allocated, one for TX and one for the RX direction. The ***Ifx_OneEyeDasPipe.h*** file can be found in the Libraries\OneEye directory.

Configuring the DPipe

A DPipe is initialized with ***Ifx_OneEyeDasPipe_stdIfDPipeInit()***. The DPipe enables access by the shell to the OneEye Pipe.

Configuring the Shell

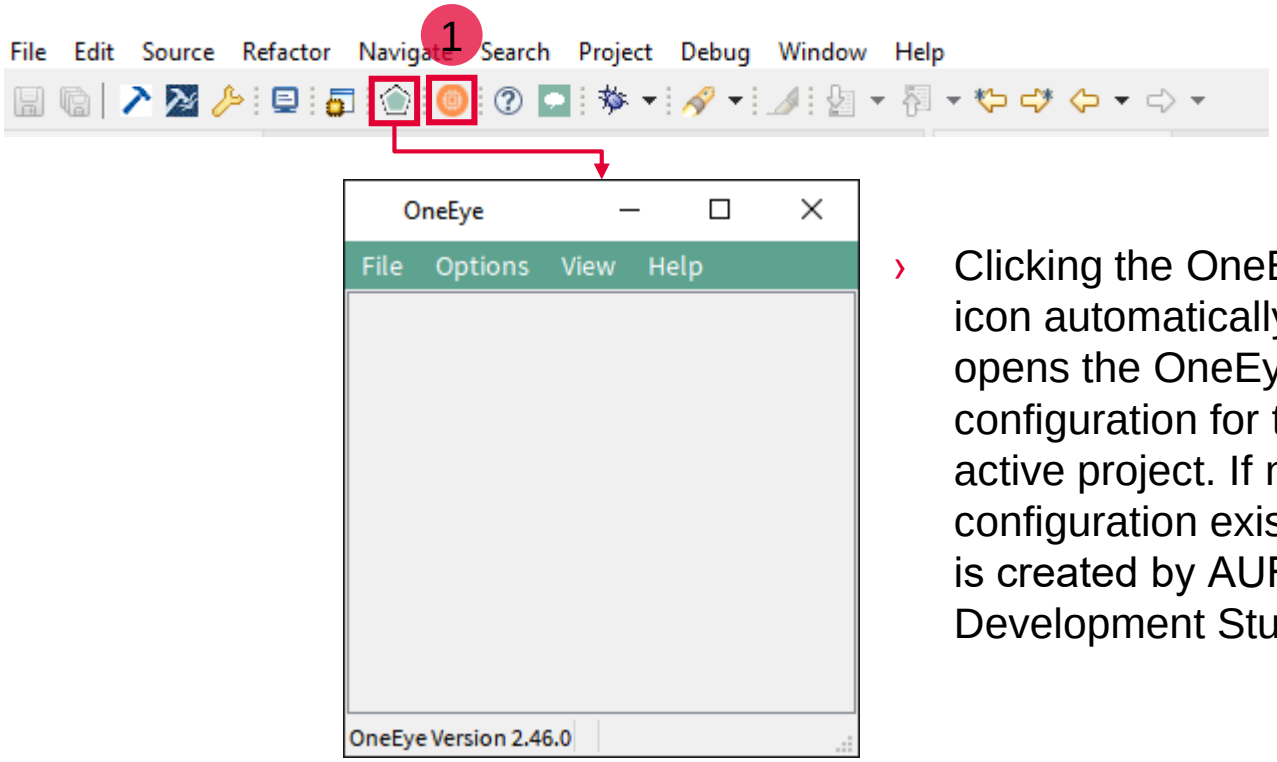
The shell is initialized with ***Ifx_Shell_init()*** passing a pointer to the previously initialized DPipe.

Running the Shell

The shell is executed in the background loop by calling ***Ifx_Shell_process()***.

Run and Test

- After code compilation, flash the device using the Flash button 1 to ensure that the program is running on the device
- For this training, the OneEye application is required for visualizing the values. OneEye can be opened inside the AURIX™ Development Studio using the following icon:



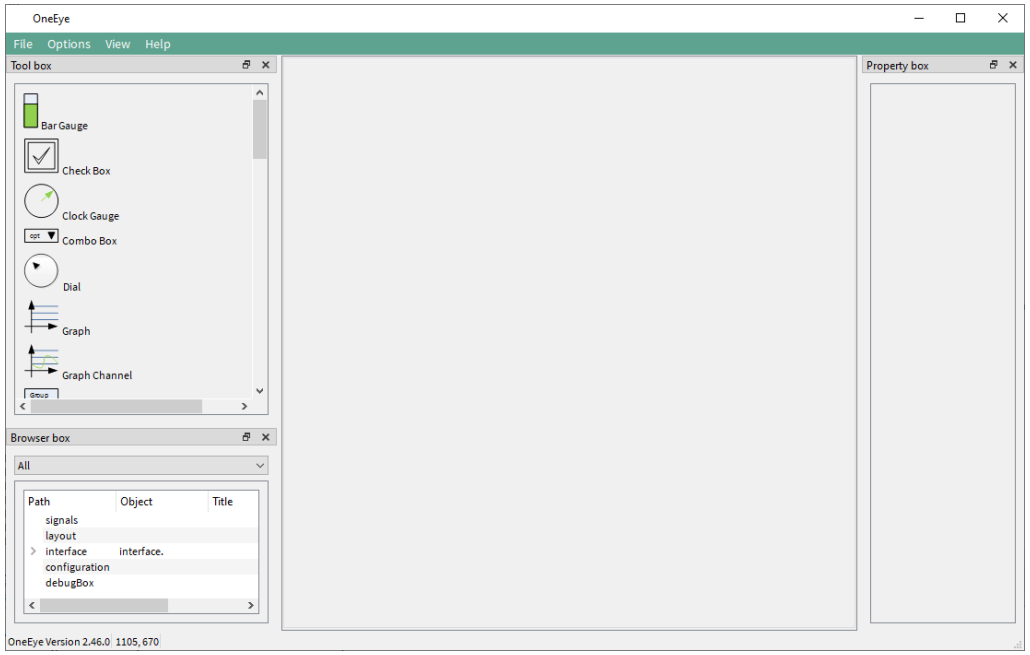
- Clicking the OneEye icon automatically opens the OneEye configuration for the active project. If no configuration exists, it is created by AURIX™ Development Studio

Implementation - OneEye

In this training, the OneEye configuration is provided inside the Libraries folder. The following steps are needed to configure the OneEye from a brand-new configuration.

Setup OneEye for editing

Select the OneEye menu “**Options -> Edit mode**” (if not already checked) to enable the edit mode.
Select the OneEye menu “**View -> Browser box**”, “**View -> Property box**” , “**View -> Tool box**” (if not already checked) to display the browser, property box, and tool box.
Close the Welcome screen if it was shown.

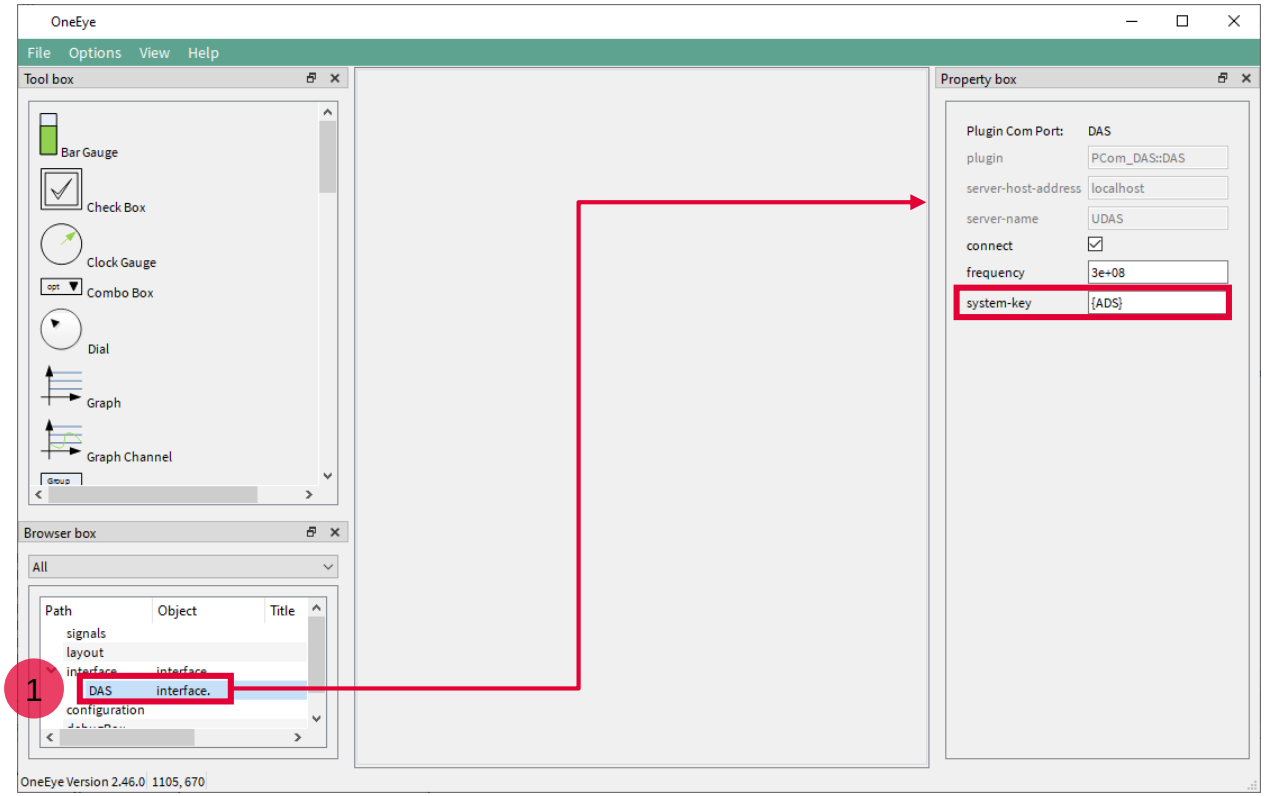


Implementation - OneEye

Configuring the DAS interface

When the OneEye configuration is created by ADS, it is already setup with a DAS interface.
Select the DAS interface in the Browser box 1.

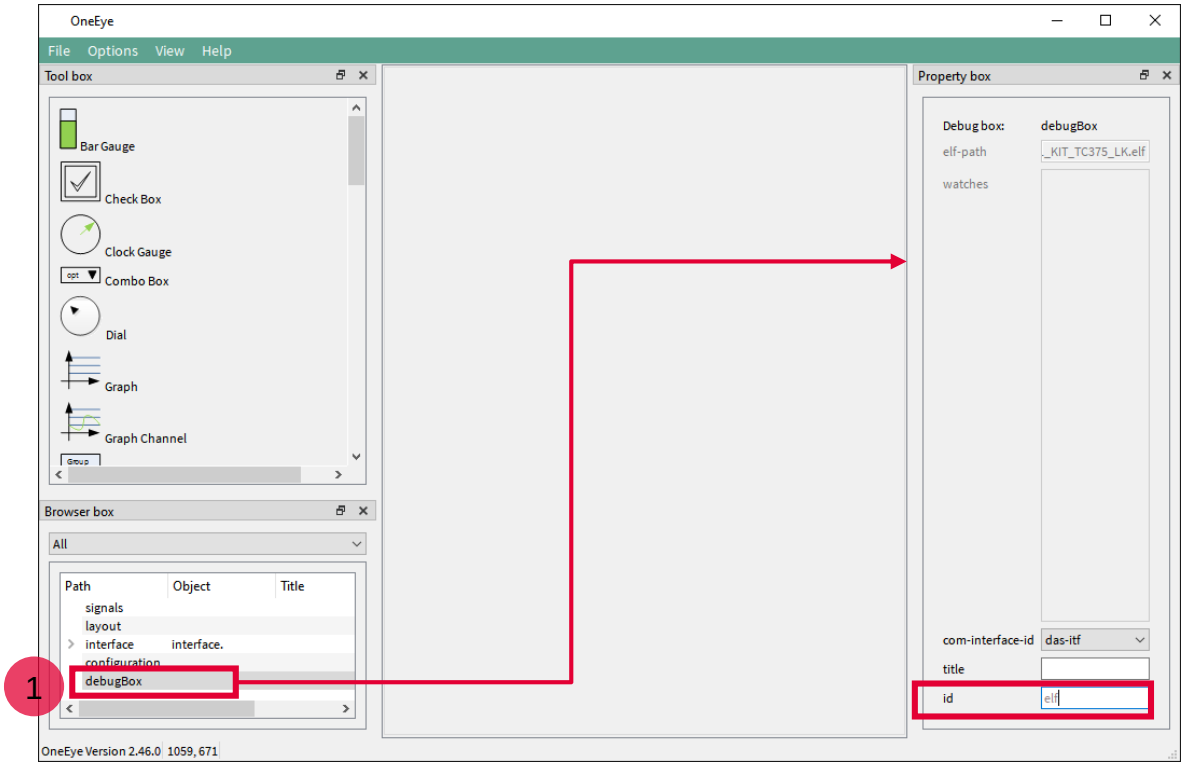
Notice the “system-key” **{ADS}** that enables the connection to the device in parallel with the ADS debugger.



Implementation - OneEye

Create a debug box to get access to variables from the .elf file

A debugBox item is already setup by default when ADS creates the OneEye configuration, preconfigured with the project .elf file path.
Select the DAS interface in the Browser box 1.
Set the id property to "**elf**", which enables to group variables into the signal tree later.



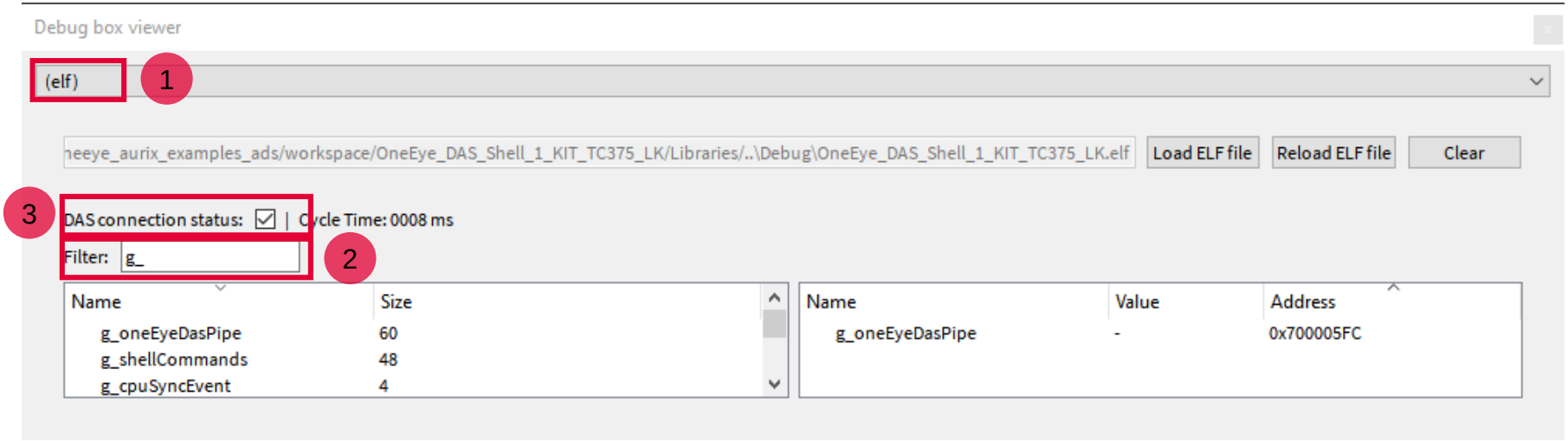
Implementation - OneEye

Open the debug box viewer and connect to the device

Select the OneEye menu “**View -> Debug box viewer**” (if not already checked) to display the debug box.
Select the debug box with the id “(elf)” 1 if not yet selected by default.
Note that the debug box enables the selection of the .elf file to be used to get information about the variables.

The **Filter** field 2, enables to filter variables by name. E.g. in this example, entering “g_” will filter for global variables.

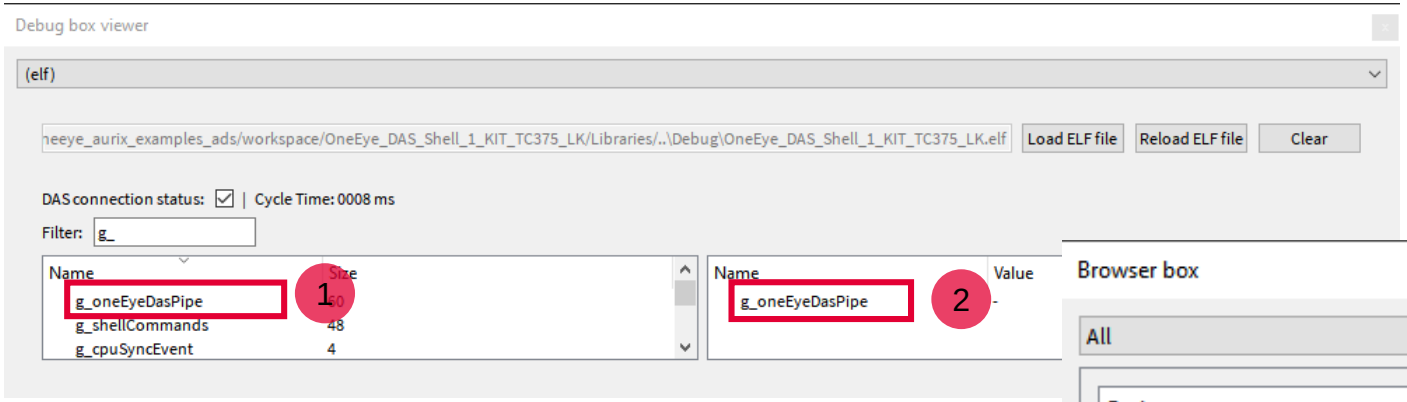
To enable the connection with the microcontroller and have read / write access to variables, check the “**DAS connection status**” box 3.

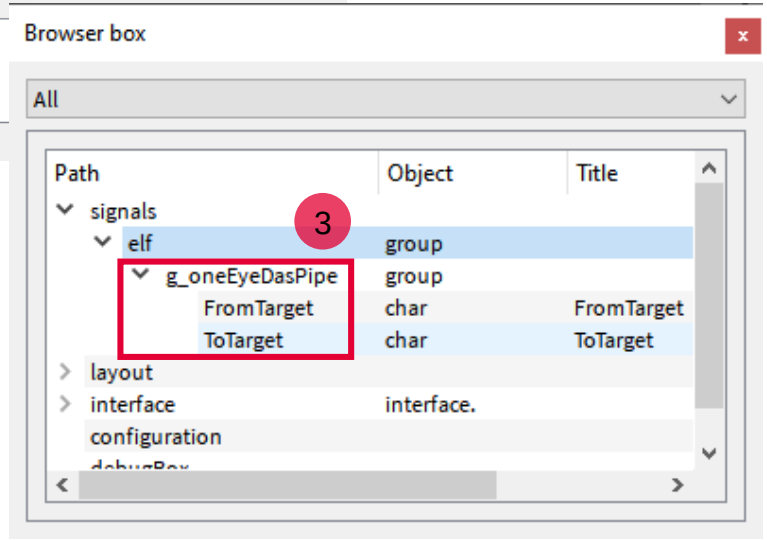


Implementation - OneEye

Create signals for the Pipe

In the debug box, search for the ***g_oneEyeDasPipe*** (1) variable, right click on it and select "**Create pipe for: g_oneEyeDasPipe**". The watch should appear on the right side of the debug box (2). Watches are periodically polled for new values on the micro controller.
Two signals are also automatically created to access the pipe (3), one for the OneEye to microcontroller direction (***ToTarget***), and one for the microcontroller to OneEye direction (***FromTarget***).



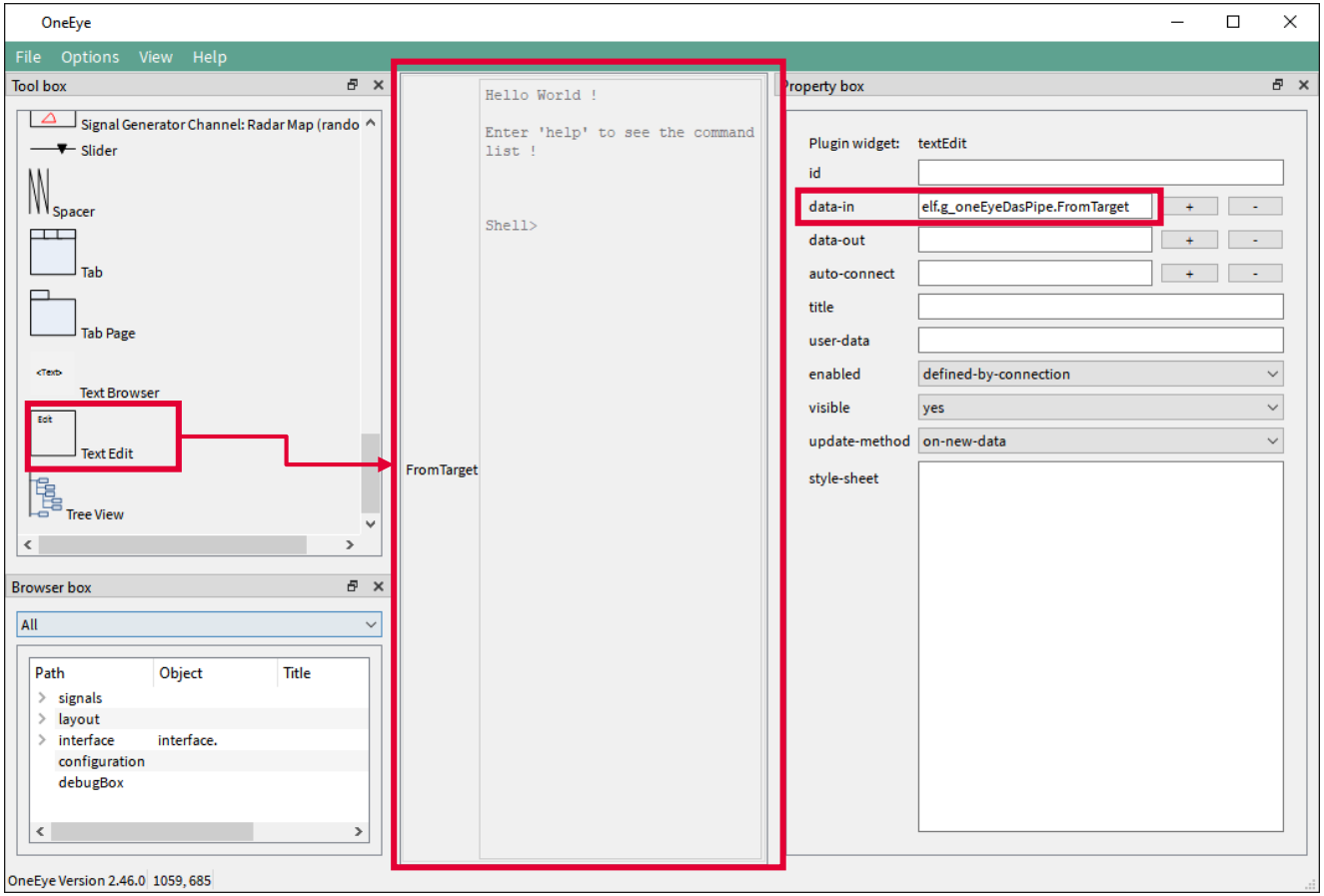


The created signals should appear in the browser box under the "signals.elf" group

Implementation - OneEye

Create a text box to display the shell text

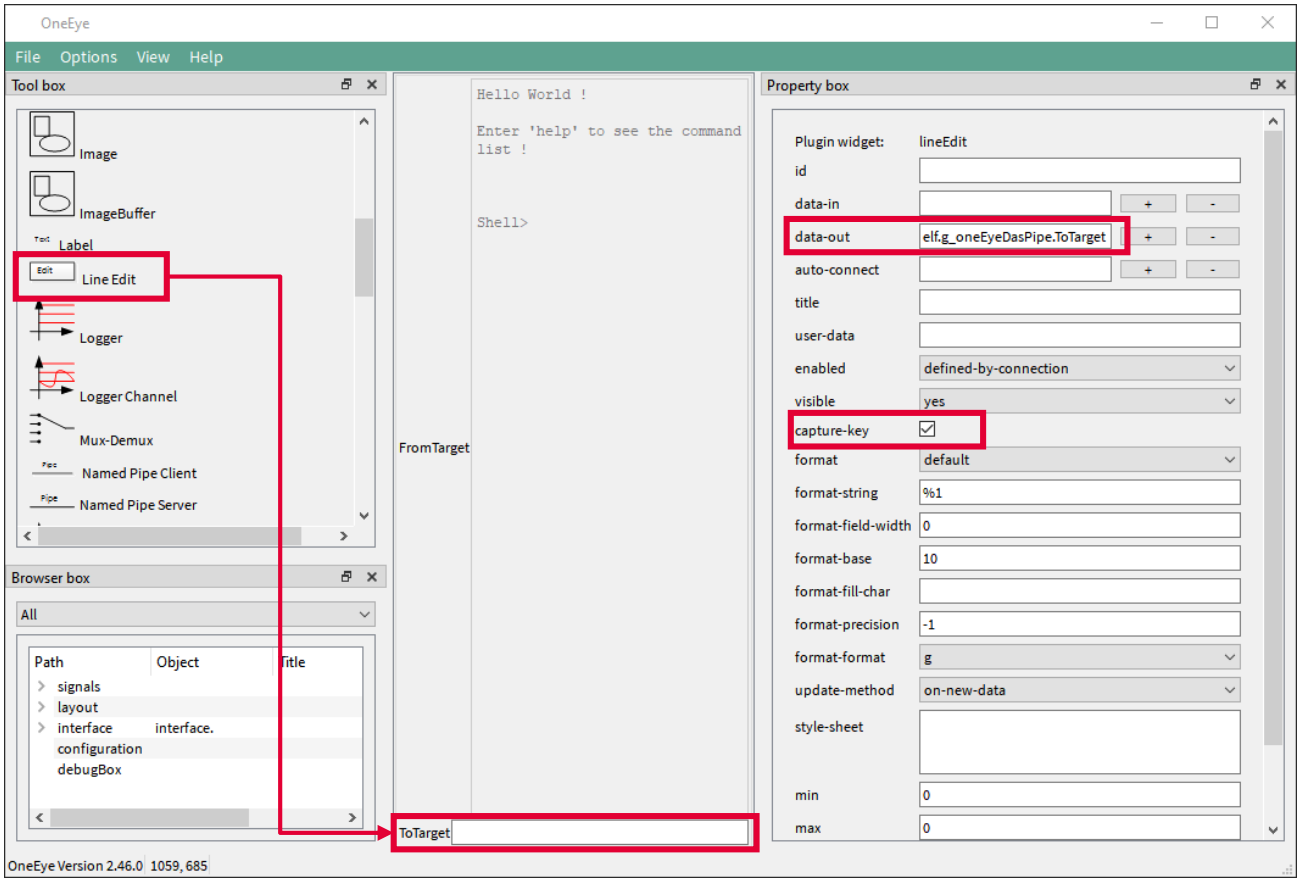
Drag and drop a **textEdit** item from the toolbox onto the layout, and set its **data-in** property to **elf.g_oneEyeDasPipe.FromTarget**.



Implementation - OneEye

Create a line edit to enter key stroke to the shell

Drag and drop a **lineEdit** item from the toolbox onto the layout, and set its **data-out** property to **elf.g_oneEyeDasPipe.ToTarget**. **Check** the **capture-key** property to enable each key stroke to be send.

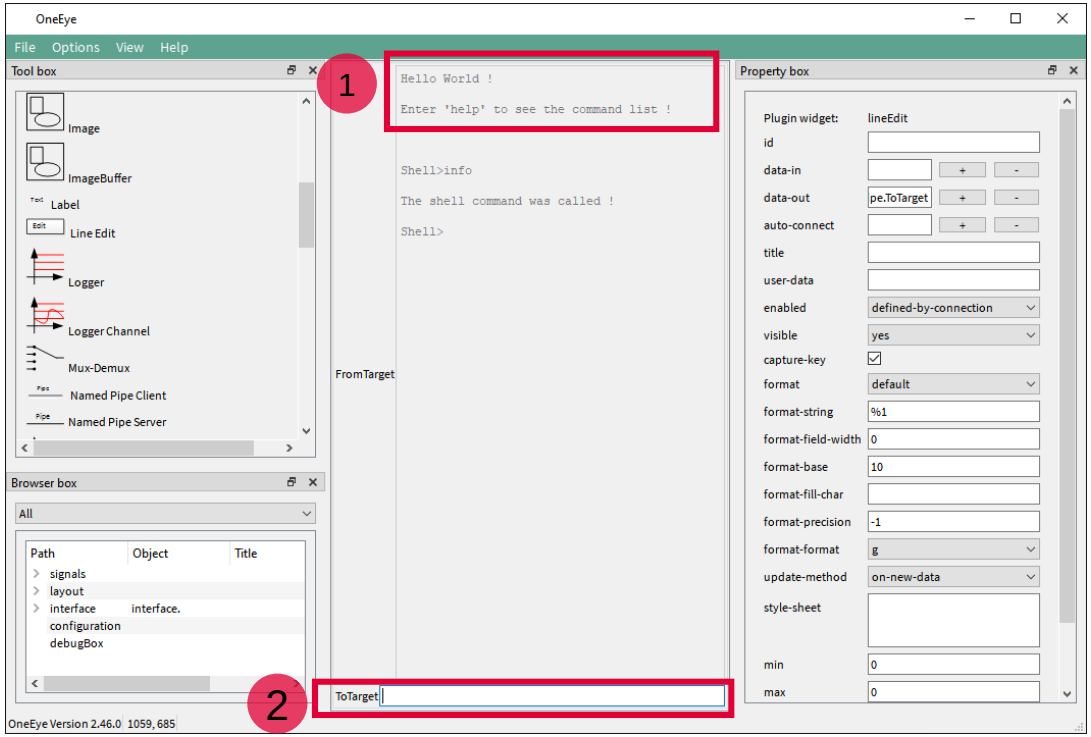


Implementation - OneEye

Test the shell interface

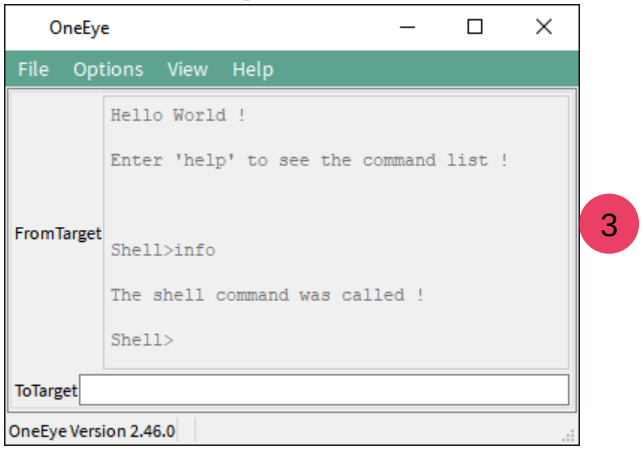
The shell textbox should display the “Hello World !” text 1, if it is not the case, check that the “**DAS connection status**” is checked in the debug box viewer.

Enter “info” in the ***ToTarget*** lineEdit field 2 and press ENTER, the microcontroller will execute the ***printShellInfo()*** function and should answer as below to acknowledge the command.



Save your configuration with CTRL+S.

Exit the edit mode with the OneEye menu “**Options -> Edit mode**” and close the “**Debug box viewer**” to only see the GUI 3.



References

- AURIX™ Development Studio is available online:
- https://www.infineon.com/aurixdevelopmentstudio
- Use the *„Import...“* function to get access to more code examples.

- More code examples can be found on the GIT repository:
- https://github.com/Infineon/AURIX_code_examples

- For additional trainings, visit our webpage:
- https://www.infineon.com/aurix-expert-training

- For questions and support, use the AURIX™ Forum:
- https://www.infineonforums.com/forums/13-Aurix-Forum

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2022-03
Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about this document?
Email: erratum@infineon.com

Document reference
OneEye_DAS_Shell_1
_KIT_TC275_LK

IMPORTANT NOTICE
The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie") .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (www.infineon.com).

WARNINGS
Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.