FAQ 748

TargetLink: How to Generate Common *.h/.c Files Used in Several CGUs?

Question

I have a .c/.h file (let's call it my_common_functions.c/h) which contains several functions and is shared across multiple CGUs (code generation units). I experience problems because multiple production code versions of this file are generated. A typical scenario for this is a Simulink library that contains multiple subsystems. Each of the subsystems has a TargetLink function block and the same module (my_common_functions) is specified.

When code for a CGU is generated, the CGU generates my_common_functions.c/h, which contains only the used functions of this CGU. This typically leads to none of the CGUs generating a complete my_common_functions.c/h.
Therefore, when building the application for the all CGUs, compiler and linker errors can occur.
How can I get a complete my_common_functions.c/h?

Solution

If a CGU uses all functions and thus generates a complete my_common_functions.c/h, you can use a ModuleOwnerShip object in the Data Dictionary and specify this CGU as the owner of the module. As a result, only this CGU generates theproduction code for my_common_functions.c/h.

The following solution applies to the general case where no CGU generates a complete my_common_functions.c/h:
The solution is to create an additional CGU, such as a separate TargetLink Subsystem (for example, called DummySubsystemForCommonFunctions), which contains all the library subsystems that you want to generate to my_common_functions.c/h.

Create a ModuleOwnerShip object in the TargetLink Data Dictionary and specify it as follows:

- SystemName = DummySubsystemForCommonFunctions
- /ModuleRefs.ModuleRef = the DD Moduel object for my_common_functions.c/h

Now, the production code version of my_common_functions.c/h is generated only when code is generated for the CGU DummySubsystemForCommonFunctions. When generating code for the other CGUs, only a stub code version of the file is generated, which contains only the content used in this specific CGU. This stub code version is ignored for the build process across all CGUs as soon as a production code version of my_common_functions.c/h. exists.

To generate the function calls in DummySubsystemForCommonFunctions, you have to make sure these functions are actually used. Otherwise, TargetLink will skip generating the functions and their implementation. You can achieve this by connecting at least one outport of each function to a root outport.

This dummy use results in an unnecessary .c/h file, which contains the call of all the functions. To avoid generating it to your production code folder, use a ModuleOwnerShip with a non-existing owner:

- SystemName = xxx_stubcode_xxx
- /ModuleRefs.ModuleRef = Root module for DummySubsystemForCommonFunctions

Note: If you do not want the common file to contain functions but variables only, use the TargetLink Code generation form the Data Dictionary feature as demonstrated in the TargetLink demo model cg_from_dd. Refer to the TargetLink user documentation for details about code generation from the Data Dictionary.

Related FAQs

- FAQ 746 – How to Configure a Reusable Subsystem and Its Identifiers for a Use in Several CGUs?
- FAQ 774 – Module Ownership, Modules, Code Generation Units, Stub Code

FAQ Overview

http://www.dspace.com/go/faq

Support

To request support, please use the form at http://www.dspace.com/go/supportrequest

Updates and Patches

Software updates and patches are available at http://www.dspace.com/go/patches.
dSPACE strongly recommends to use the most recent patches for your dSPACE installation.

Important Notice

dSPACE GmbH
Rathenaustraße 26
33102 Paderborn
Germany

This publication and the contents hereof are subject to change without notice.

A list of registered dSPACE trademarks is available at: http://www.dspace.com/go/Trademarks