SIEMENS

FAQ • 10/2016

Know-how Protection in Fail-safe Programs

STEP 7 Safety Basic/Advanced V14

https://support.industry.siemens.com/cs/ww/en/view/109742314

This entry is from the Siemens Industry Online Support. The general terms of use (http://www.siemens.com/terms_of_use) apply.

Security Information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.
In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.
Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.
Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit http://www.siemens.com/industrialsecurity.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.
To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under http://www.siemens.com/industrialsecurity.

Contents

1 Introduction

Validity

The information in this document applies for Safety Advanced V14 and Safety Basic V14 and higher.

Know-how protection

A know-how protection for one or multiple program blocks protects against unauthorized access. This can be a useful requirement in particular for fail-safe blocks of a safety program.

This document includes instructions for handling a know-how protection option for fail-safe blocks.

In the following the designation "blocks" always refers to F blocks (F-FCs or F-FBs).

2 Creating and Removing Know-how Protection

2.1 Creating Know-how Protection

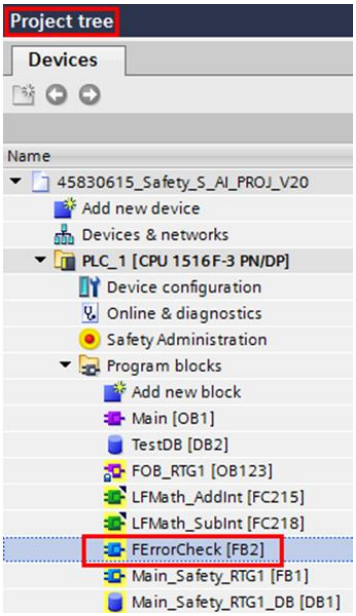
Requirements for creating a know-how protection:

- The safety program uses the blocks to be encrypted.
- The safety program is consistent.

You can create the know-how protection for one or multiple blocks. Proceed as follows.

- Mark the block(s) in the project tree of STEP 7, for example.

Figure 2-1 Marked F block in the project tree



- Menu "Edit > Know-how Protection" (or via right mouse button).
- In the window that opens you click "Define", create a new password and confirm with "OK".

After creating a know-how protection for a block you can no longer view the block in the TIA Portal. The associated backup blocks created by the system in the system block folder are encrypted with the user password and are no longer visible to the system.

2.2 Removing Know-how Protection

The block editor must be closed to be able to remove the know-how protection.

Proceed as follows to remove the know-how protection:

- Mark the block(s) in the project tree of STEP 7, for example.
- Menu "Edit > Know-how Protection" (or via right mouse button).
- In the window that opens you deselect "Do not show code (know-how protection)", enter the password and confirm with "OK".

If you select multiple know-how-protected blocks and the item "Know-how Protection" is not available, one of the editors of the blocks to be unblocked is still open.

2.3 Temporarily Removing Know-how Protection

When making changes to the safety program you should completely remove the know-how protection from all blocks and create it again after the changes have been made. When making minor changes in F-FCs and F-FBs it is useful to just temporarily remove the know-how protection. Minor changes include, for example, a change of the logic without affecting global access.

Proceed as follows to temporarily remove the know-how protection:

- Double-click a know-how-protected block in the project tree, for example.
- Enter the password and confirm with "OK".

The project tree shows the block as know-how protected, however the block is opened for editing in the editor. Closing the editor re-activates the know-how protection. After closing, with "Edit > Undo" the password would be demanded for unblocking the block.

Note After changing a block whose know-how protection has been temporarily removed you should first compile the safety program and then close the block. Otherwise you get a password query during compilation.

3 Copying and Inserting Know-how-protected Blocks

3.1 Copying Single Know-how-protected Blocks

Requirement: The blocks to be copied do not call any other blocks (FCs or FBs). If this requirement is fulfilled, you can copy the know-how-protected blocks to the destinations below:

- Other fail-safe SIMATIC S7 controllers
- Master copies of the project library
- Master copies of the global library

Note Copying a know-how-protected block from an S7-121xF controller to an S7-151xF controller leads to a decryption request because different controller families are involved.

If you copy within the same controller family (for example from an S7-1515F to an S7-1518F or from an S7-121xF to an S7-121xF) you do not get a decryption request.

3.2 Copying Call Sequences of Know-how-protected Blocks

Observe the points below for copying call sequences of know-how-protected blocks:

- Only multi-instances may be used in the call sequences.
- Call sequences must be copied completely. In the target controller the compiler receives notification of missing blocks in order to be able to make a consistent compilation.
- Know-how-protected blocks should not call any unprotected blocks in such sequences. Otherwise, when inserting in other controllers, the blocks to be inserted might be renamed in their programs. This can lead to a password request for the know-how-protected block during compilation.

Note Create a separate group (folder) for a call sequence in the program block folder. Group all the dependent blocks there. Then you can simply use this folder to copy all the blocks to the other controller instead of via a multiple selection.

Example of a call sequence

- F-FB "first" [FB2] calls F-FB "second" [FB3] as multi-instance.
- F-FB "second" [FB3] calls block ESTOP1 as multi-instance.

The F-FB "first" is called as instance (instDB) in the Main Safety of runtime group 1. The blocks F-FB "first" and F-FB "second" are know-how protected and are in the "myLib" folder.

Instead of copying the blocks individually you simply insert the "myLib" folder into the target controller. The instance DB of the F-FB has to be newly created on the target controller when the F-FB is called in the Main Safety and must not be copied along from the source controller.

3.3 Transferring Know-how-protected Blocks with Master Copies

Using individual blocks without call sequences is not restricted in any way worth mentioning.

The following conditions have to be met for saving call sequences that are to be copied into the project or global master copies:

- Exclusive use of multi-instances in the call sequences
- No use of cross access to instance data blocks of other blocks.
- Call sequences are complete.
- Only know-how-protected F call sequences.

Important: If a call sequence is to be stored in a master copy, it has to be inserted as a group. Therefore, that which is a recommendation for copying and inserting from one controller to another is mandatory for use in master copies.

3.4 Copying a Controller

In addition to copying and inserting blocks it is also possible to copy a fail-safe controller between two TIA Portal projects or forward a fail-safe controller to a master copy. The special features for using know-how-protected blocks mentioned in this chapter do not apply here.

4 More Help for Creating Libraries

The information in chapter 3 is addressed to creators of libraries. They have to make sure that the user of a library procured as master copy does not received a decryption request.

This chapter continues in the same vein and provides more help for library creation options:

- Provision of independent single blocks (FCs/FBs).
- Provision of a subprogram with interdependent FBs and FCs.

4.1 Providing Independent Single Blocks (FCs/FBs)

If you want to provide single know-how-protected blocks, you just have to fulfill the following conditions: System library blocks called internally (B. ESTOP1, for example) have to be called as multi-instances.

User documentation produced for the library must include the following:

- Utilization and parameterization of the blocks.
- Interconnection of the blocks.
- PLC data types and variables used in the program are from the master copy.

4.2 Providing a Subprogram with Interdependent FBs and FCs.

How to forward encrypted F call sequences via a master copy has already been explained in chapter 3. This procedure is illustrated in the example below.

Example

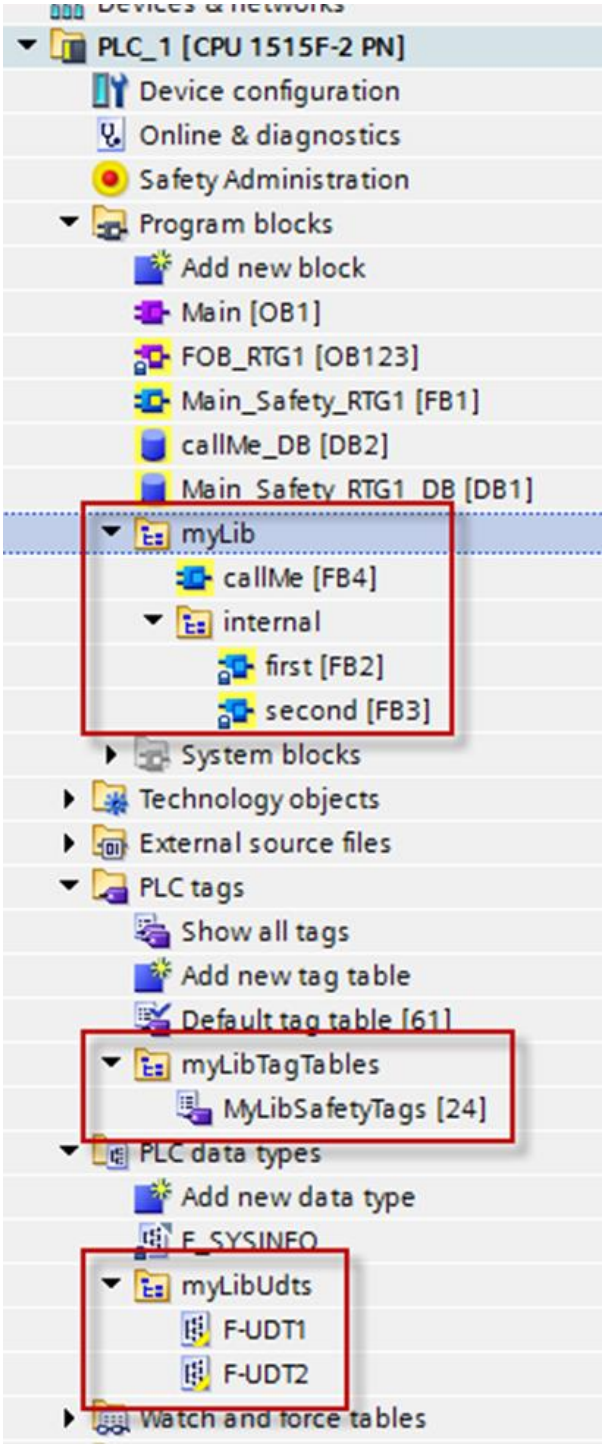
Creating blocks for forwarding is identical to creating a safety program. Here you should comply with the recommendations given in the programming guidelines (https://support.industry.siemens.com/cs/ww/en/view/90885040):

- No access (read or write) to global DBs.
- Forwarding of data between blocks via block interfaces.
- Use of F-compliant user data types for clarity of the interfaces.

The blocks for forwarding are grouped in a separate folder ("myLib", for example) as shown in the next figure. This can be further subdivided. The block called by the user is only in the topmost folder (myLib) and all other dependent blocks are in another subfolder (internal, for example). For the sake of clarity it is recommended to use this procedure also for user-defined F-compliant user data types and variables.

To be able to create know-how protection for the blocks you have to use them in the safety program and this must be compiled and consistent.

Figure 4-1 Separate folder for the blocks to be forwarded



Move the topmost folder to the project master copy or to a global master copy. If there are dependent, F-compliant user-defined user data types and variables, copy these likewise to the same master copy (see figure below).

Figure 4-2 Copying F-compliant user-defined user data types and variables