How to make a Seed & Key.DLL for *X*CP

Version 1.0

Modification protocol:

Document Version	Author	Date	Content (Detailed Modification Protocol)
V1.0	Lr	16.09.02	First version

Content

1. Introduction to Seed & Key

The *X*CP specification provides a security mechanism called Seed & Key. Hereby the slave requests a key before granting access. The master must ask the slave for a seed and compute the respective key.

In order to keep the algorithm for calculating the key confidential, it is not known to the master.

The confidential algorithm is encapsulated in a Seed & Key.DLL. This document describes a template to build a Seed & Key.DLL as to be used for *X*CP.

2. How to make a Seed & Key.DLL

2.1. *Files*

The following files are required:

- Callconv.h
- seedNKeyXcp.h
- seedNKeyXcpMain.cpp
- SeedNKeyXcp.dsp
- How to make a Seed & Key.DLL for XCP.doc (this document)

2.2. *Project*

SeedNKeyXcp.dsp contains a entirely setup project for Microsoft VisualC++ V6.0.

2.3. *Algorithm*

The function **computeKeyFromSeedxxx(...)** must be filled with the actual algorithm. A simple copy of the seed into the key is implemented as example. The example assumes different algorithms for each privilege. Hence there are functions for each access privilege:

-	computeKeyFromSeedCalPag(...)	for Calibration access and Paging access,
-	computeKeyFromSeedDaq(...)	for Acquisition access,
-	computeKeyFromSeedStim(...)	for Stimulation access,
-	computeKeyFromSeedPgm(...)	for Programming access.

The algorithm must be the same than the one implemented in the slave.
Please be aware that most slaves are using Motorola byte order whereas the algorithm in the SeedNKey-DLL runs on an Intel CPU.

2.4. *Privileges*

A SeedNKey-DLL can grant several privileges as defined in the XCP specification. In function **setMyPrivilege()** you can determine which privileges the SeedNKey-DLL will provide. In the example all available privileges are set for your convenience. So you can easily comment out the ungranted privileges.
The example SeedNKey-DLL will not compute a key for unset privileges. Hence you can actually use the same algorithm for all privileges and only change the privilege settings to build DLLs granting different privileges.

2.5. *Building the DLL*

Before makeing changes you should try to build a SeedNKey-DLL to ensure that it works to build the example. The SeedNKey-DLL will be written to the respective subdirectory (Release / Debug).