BOSCH
Invented for life

Comparison of VHDL-AMS and Modelica

inside.Docupedia Export

Author: Hofmann Andreas (DC/ESS12)
Date: 05-Nov-2018 10:18

Table of Contents

Draft

This only gives a rough overview about the encryption standards defined for Modelica and VHDL-AMS.

This document has not yet been released!

1 Motivation:

Within the IEEE Standard 1076.1 (Revision 2017) the Extension Analog and Mixed-Signal (VHDL-AMS) of the IEEE Standard VHDL is defined. It also contains information on encryption respectively protection of intellectual property.
Since VHDL-AMS is a modeling language with many similarities to the Modelica modeling language approaches for IP protection should be compared.

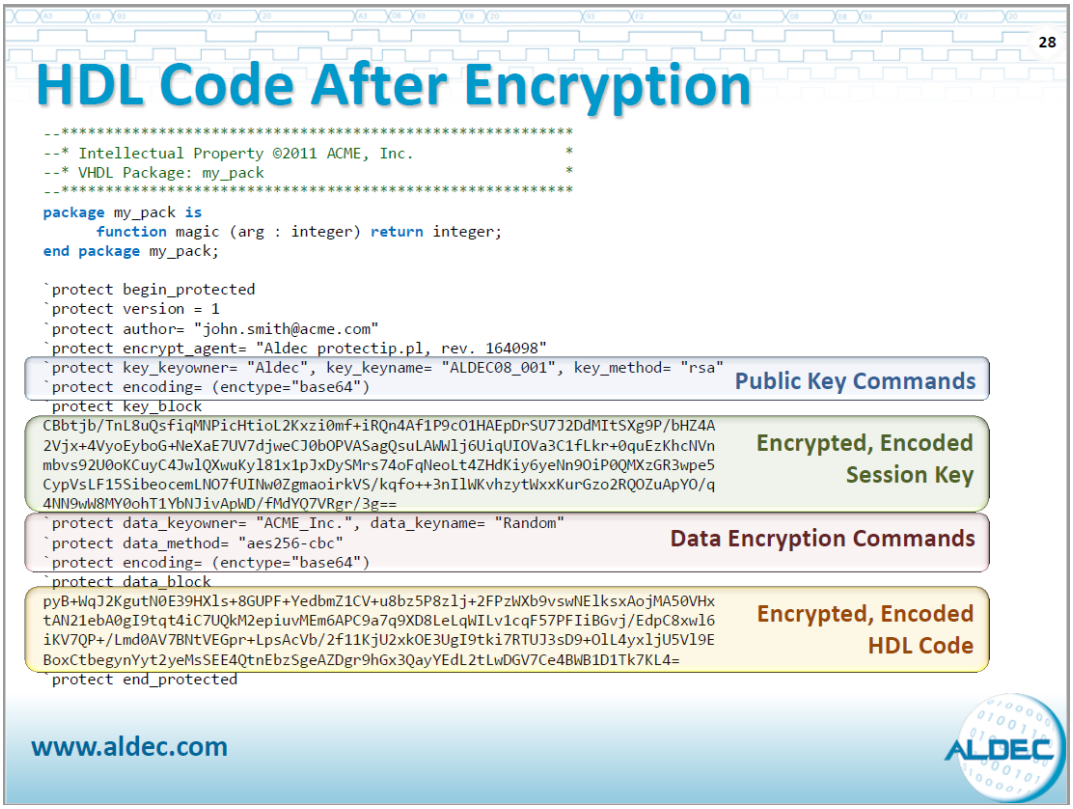
	VHDL-AMS	Modelica Specification (3.4)	Modelon Proposal for Licensing and Encryption (Modelica ticket 1868)
Protection units	Encryption of portions of a VHDL design file using keypwords: **`protect begin**, **`protect end**	Protection of full classes and its childen (hierarchically on a lower level) by protection annotation valid for whole class	Container (zip-archive) with directory structured modelica model containing encrypted models (*.moc) and non-encrypted models (*.mo) with protection annotation
Encryption methods for protecting source code	Different symmetric and asymmetric algorithms for encryption of the code (DES/3DES, AES, RSA, PGP, ...)	not defined in standard	Encryption of source code is provided by library vendor specific tool
Signature of code/ Protection against altering of code	Different functions to create hash of original source code before encryption to detect possible altering of file after decryption (digest methods)	not defined in standard	not considered
Encoding of encrypted part (byte sequence)	Different defined encoding types to prevent modification by different text editors	Pictures are stored in Base64 encoding UTF-8 is used as text encoding for files	not considered

	VHDL-AMS	Modelica Specification (3.4)	Modelon Proposal for Licensing and Encryption (Modelica ticket 1868)
De/ Encryption of source code	Symmetric key only: • source code is encrypted using the symmetric key • key needs to be provided manually on a different channel Asymmetric key only: • source code is encrypted using the public keys of the tools that should be supported • the tools can decrypt the source code using their private keys Combination of symmetric and asymmetric keys: • a random session key is generated and used as symmetric encryption key • the session key is then encrypted using the public key of all tool that should be supported • Decryption is done within the supported tool through the private key **→ Public keys must be available to the encrypting tool**	not defined in standard	First of all a TSL connection (asymmetric encryption session) between the Modelica tool and the library vendor specific tool is established. Within the session the library vendor specific tool encrypts all data with the public key of the Modelica tool, which can finally decrypt the data stream using its private key. (Part of TLS) Encryption of the source code is done within the library vendor specific tool, therefore no exchange of key is needed. All Modelica tools, which public keys are known to the library vendor specific tool can use the encrypted library, provided the license is valid. **→ Public keys of the Modelica Tools must be available to the library vendor specific tool.** **It is not known which encryption/decryption method is used.**
Encryption of encrypted models	Since the encryption of a model is done only internally, it is possible to stack multiple encryption layers above each other.	Not allowed w.r.t. specification Encrypted packages are defined as read only, therefore encryption of an encrypted file is not relevant/possible.	Not allowed w.r.t. specification

	VHDL-AMS	Modelica Specification (3.4)	Modelon Proposal for Licensing and Encryption (Modelica ticket 1868)
Code Generation (Compilation of encrypted models)	not defined in standard	not defined in standard	dependent on the Modelica compiler
Licensing	Not defined in specification	Licensing is defined using license files/keys for the encrypted Modelica library. However, requires encrypted libraries and working encryption in general	Custom licensing can be archived using the library vendor specific tool. However, due to the encryption it should also be possible to use the licensing defined within the Modelica specification
Benefits	• No additional tools required (all relevant information regarding encryption stored in model itself) • Protection against altering of files	• licensing mechanism defined in specification	• Arbitrary licensing mechanism possible • Common, standardized interface for all tools
Drawbacks	• No licensing mechanisms • Public keys must be available to all tool vendors • Used tool must support type of encryption	• specification does not define encryption implementations	• Library vendor specific tool must be developed/ maintained by library vendor or licensed • Public keys must be available to everyone (central trustworthy source) • Technical details not yet fully available (e.g. handshake and communication between tools)

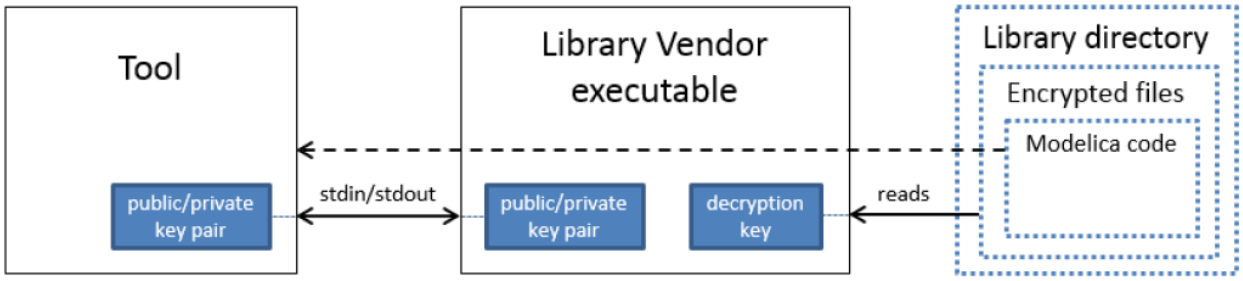
2 Example for encryptions:

2.1 VHDL-AMS:



Source: Kaczynski, J.: Decrypting Encryption in HDL Desing and Verification. ALDEC Webinar.
Link: Online (last accessed: 26.09.2018)

2.2 Modelon Proposal for Licensing and Encryption:



Source: Mattsson, et al.: Suggestion for a standard for encryption, licensing and packeting of libraries
Link: Online (last accessed: 26.09.2018)