



**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

<b>1</b>	<b>Motivation:</b>	<b>4</b>
<b>2</b>	<b>Example for encryptions:</b>	<b>7</b>
2.1	VHDL-AMS:	7
2.2	Modelon Proposal for Licensing and Encryption:	7

**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 keywords: <b>`protect begin, `protect end</b>	Protection of full classes and its children (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	<p>Symmetric key only:</p> <ul style="list-style-type: none"> <li>source code is encrypted using the symmetric key</li> <li>key needs to be provided manually on a different channel</li> </ul> <p>Asymmetric key only:</p> <ul style="list-style-type: none"> <li>source code is encrypted using the public keys of the tools that should be supported</li> <li>the tools can decrypt the source code using their private keys</li> </ul> <p>Combination of symmetric and asymmetric keys:</p> <ul style="list-style-type: none"> <li>a random session key is generated and used as symmetric encryption key</li> <li>the session key is then encrypted using the public key of all tool that should be supported</li> <li>Decryption is done within the supported tool through the private key</li> </ul> <p>→ <b>Public keys must be available to the encrypting tool</b></p>	not defined in standard	<p>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)</p> <p>Encryption of the source code is done within the library vendor specific tool, therefore no exchange of key is needed.</p> <p>All Modelica tools, which public keys are known to the library vendor specific tool can use the encrypted library, provided the license is valid.</p> <p>→ <b>Public keys of the Modelica Tools must be available to the library vendor specific tool.</b></p> <p><b>It is not known which encryption/decryption method is used.</b></p>
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.	<p>Not allowed w.r.t. specification</p> <p>Encrypted packages are defined as read only, therefore encryption of an encrypted file is not relevant/possible.</p>	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	<ul style="list-style-type: none"> <li>No additional tools required (all relevant information regarding encryption stored in model itself)</li> <li>Protection against altering of files</li> </ul>	<ul style="list-style-type: none"> <li>licensing mechanism defined in specification</li> </ul>	<ul style="list-style-type: none"> <li>Arbitrary licensing mechanism possible</li> <li>Common, standardized interface for all tools</li> </ul>
Drawbacks	<ul style="list-style-type: none"> <li>No licensing mechanisms</li> <li>Public keys must be available to all tool vendors</li> <li>Used tool must support type of encryption</li> </ul>	<ul style="list-style-type: none"> <li>specification does not define encryption implementations</li> </ul>	<ul style="list-style-type: none"> <li>Library vendor specific tool must be developed/ maintained by library vendor or licensed</li> <li>Public keys must be available to everyone (central trustworthy source)</li> <li>Technical details not yet fully available (e.g. handshake and communication between tools)</li> </ul>

## 2 Example for encryptions:

### 2.1 VHDL-AMS:

**HDL Code After Encryption**

```

--*****
--* Intellectual Property ©2011 ACME, Inc.          *
--* VHDL Package: my_pack                          *
--*****
package my_pack is
    function magic (arg : integer) return integer;
end package my_pack;

`protect begin_protected
`protect version = 1
`protect author= "john.smith@acme.com"
`protect encrypt_agent= "Aldec protectip.pl, rev. 164098"
`protect key_keyowner= "Aldec", key_keyname= "ALDEC08_001", key_method= "rsa"
`protect encoding= (enctype="base64")
`protect key_block
CBbtjb/TnL8uQsfiqMNPicHtiol2Kxzi0mf+iRQn4Af1P9c01HAEPDrSU7J2DdMItSXg9P/bHZ4A
2Vjx+4VyoEyboG+NeXaE7UV7djweCJ0b0PVASagQsuLAWMlj6UiqUI0Va3C1fLkr+0quEzKhcNvn
mbvs92U0oKcuyC4Jw1QXwuKy181x1pJxDySMrs74oFqNeolt4ZHdKiy6yeNn90iP0QMxzGR3wpe5
CypVsLF155ibeocemLNO7FUIInw0ZgmaoirkVS/kqfo++3nI1NKvhzyTwxKurGzo2RQ0ZuApY0/q
4NN9w8MY0ohT1YbNjivApwD/fMdY07VRgr/3g==
`protect data_keyowner= "ACME_Inc.", data_keyname= "Random"
`protect data_method= "aes256-cbc"
`protect encoding= (enctype="base64")
`protect data_block
pyB+WqJ2KgutN0E39HX1s+8GUPF+YedbmZ1CV+u8bz5P8zlj+2FPzWXB9vswNE1ksxAojMA50VHx
tAN21ebA0gI9tqt4iC7UqkM2epiuvME6APC9a7q9XD8LeLqWILv1cqF57PFi8Gvj/EdpC8xw16
iKV7QP+/Lmd0AV7BNtVEGpr+LpsAcVb/2f11KjU2xk0E3UgI9tki7RTUJ3sD9+01L4yx1J5V19E
BoxCtbeqynYyt2yeMsSEE4QtnEbZSgeAZDgr9hGx3QayYEdL2tLwDGV7Ce4BWB1D1Tk7KL4=
`protect end_protected
    
```

**Public Key Commands**

**Encrypted, Encoded Session Key**

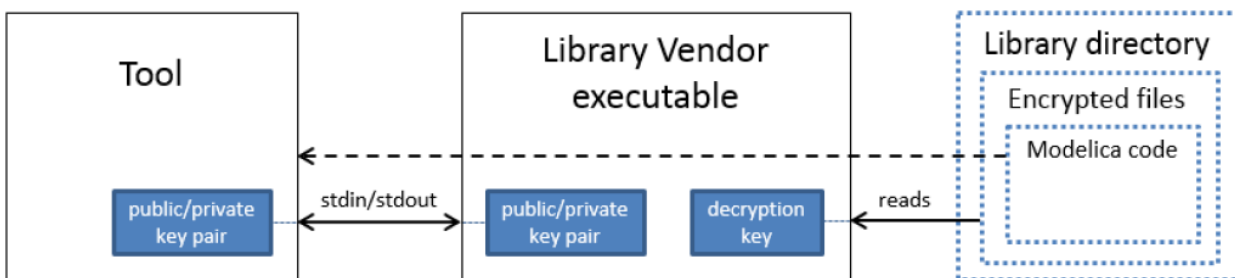
**Data Encryption Commands**

**Encrypted, Encoded HDL Code**

www.aldec.com

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 packaging of libraries  
 Link: [Online](#) (last accessed: 26.09.2018)