Protecting Your IP Cores – Part I Soft IP, Section One: Encryption of HDL Codes

IEEE Std 1735 2014 IEEE Recommended Practice for Encryption and Management of IP HDL Source Code Protection

JAASKELAINEN2018 HDL SOURCE CODE PROTECTION

IEEE Std 1735 2014 IEEE Recommended Practice for Encryption and Management of Electronic Design Intellectual Property (IP) .

IEEE 1735 standard defines a convenient method for protecting RTL source code.

Main Goal: Interoperability across the EDA tools
Source code protection has been tested with multiple EDA tools.

How can I encrypt a Verilog / VHDL file so it can be simulated or synthesized without revealing the de-crypted source file according to IEEE 1735 standard?

The answer to that depends on which tools you want to be able read the encrypted source directly without revealing the original source and which encryption formats they support. Many tool vendors now support the IEEE 1735-2014 encryption recommendations that enables one vendor to encrypt, while another vendor de-crypts the same design. If the vendor does not support that standard, you will have to contact that vendor for their encryption tool.

Verilog / VHDL encryption is a part of the 2005 standard, and many vendors now use it. You will need a tool that will encrypt the Verilog using this standard, and most of the large simulators will do it- Modelsim, IUS, and VCS. You can refer to the individual tool's documentation for more information on that.

List of some commonly used IEEE 1735 compatible EDA tools:

Cadence
Incisive, Xcelium , Genus

Intel Altera
Quartus

Siemens Mentor Graphics
QuestaSim, ModelSim

Xilinx
Vivado

IP Core Protection with EDA and FPGA - a Proposal

Here is a proposal about IP protection with EDA and FPGA design.

First, it is assumed that the EDA software is trusted. The IP core designed by a third party must first purchase an RTL-level IP core from the developer of the EDA software. The third party encrypts the IP core with the public key of the EDA developer, and then uses the public key provided by the designer. In this way, although the designer has obtained a third-party IP core, he does not know the private key of the EDA software developer, so the designer cannot know the RTL-level description of the IP core.

The designer can treat the purchased IP core as a black box with known functions. After completing the circuit description using the black box and other circuit modules, the designer imports the design into the **EDA tool**. The EDA tool uses the key entered by the designer and the key of the EDA developer to decrypt the encrypted IP core twice to obtain a complete RTL representation of the entire design. Then, EDA tools can perform behavior synthesis, logic synthesis, and technology mapping on the design, and finally generate a netlist after technology mapping. It is assumed that the tools used for synthesis and technology mapping are provided by an EDA developer, and the FPGA placement and routing tools are provided by another EDA developer, for example, provided by the FPGA manufacturer.

Before writing the optimized gate-level netlist to a file, the **EDA tool** first encrypts the netlist with its own private key, and then encrypts the netlist with the public key of the place and route tool provider. In this way, the designer still cannot view the netlist after synthesis and technology mapping, thereby preventing the designer from using reverse engineering to steal the IP core.

The **place and route tool** first decrypts the encrypted netlist with its own private key, and then uses the public key of the front-end EDA tool developer for the second decryption to obtain the decrypted netlist. Using this netlist, the layout tool can complete the layout of the design on the chip, and finally generate a bit-stream file for FPGA configuration.

Notes:

This text is part of the book chapter named "*Part 17 IP Core Protection*", which is again part of the book titled "*Silicon IP – More than just Design*" to be prepared by Mark Chen. You may also find some little more information on the following website http://www.angelia.eu.org/.

保护您的 IP 内核——第一部分软 IP，第一节：HDL 代码的加密

IEEE Std 1735 2014 IEEE IP HDL 源代码保护加密和管理推荐实践

IEEE Std 1735 2014 IEEE 电子设计知识产权 (IP) 加密和管理推荐实践。

IEEE 1735 标准定义了一种方便的方法来保护 RTL 源代码。

主要目标：跨 EDA 工具的互操作性
源代码保护已经过多种 EDA 工具的测试。

如何加密 Verilog / VHDL 文件，以便在不泄露根据 IEEE 1735 标准的解密源文件的情况下对其进行仿真或合成？

答案取决于您希望哪些工具能够直接读取加密源而不泄露原始源以及它们支持哪些加密格式。许多工具供应商现在支持 IEEE 1735-2014 加密建议，使一个供应商能够加密，而另一个供应商解密相同的设计。如果供应商不支持该标准，您必须联系该供应商以获取他们的加密工具。

Verilog/VHDL 加密是 2005 标准的一部分，现在许多供应商都在使用它。您将需要一个使用此标准加密 Verilog 的工具，大多数大型模拟器都会这样做 - Modelsim、IUS 和 VCS。您可以参考各个工具的文档以获取更多信息。

一些常用的兼容 IEEE 1735 的 EDA 工具列表：

节奏
Incisive, Xcelium，属

英特尔 Altera
石英

西门子 Mentor Graphics
QuestaSim，模型模拟

赛灵思
Vivado 维瓦多

ALDEC 加密工具：protectip /Active-HDL / Riviera-PRO

https://www.aldec.com/en/support/resources/documentation/articles/1650
https://firsteda.com/news/how-to-secure-your-ip-with-encryption/

Aldec 支持 VHDL 和 Verilog 的加密。它在 Verilog-2005 中被添加到 Verilog 标准中，并于 2008 年成为 VHDL 的一部分。两种 HDL 加密方法相似，因为它们使用非对称加密，即您需要公钥来加密和私钥来解密。不用说，您不应该泄露您的私钥。

ALDEC 为加密工具提供了名为 protectip 的可执行文件，位于模拟器安装的 bin 文件夹中。您可以在 Active-HDL 或 Riviera-PRO 中从控制台运行它。要获取 Aldec 公钥，请参阅 Riviera-PRO 或 Active-HDL 用户手册中的“ALDEC 公钥”。但是，如果您使用 Aldec IP 加密工具，则无需指定公钥。如果由 ALDEC 工具加密的代码也将由其他工具使用，则应在 ALDEC 实用程序加密之前将包含其他工具公钥和匹配标签的适当部分添加到源中。由于 ALDEC 无权分发属于其他供应商和工具的公钥，因此 IP 创建者应从这些工具的当前文档中检索它们或联系

工具供应商以获取它们。

Cadence VHDL 源代码加密 - Incisive, Xcelium, Genus

https://community.cadence.com/cadence_technology_forums/f/functional-verification/13649/vhdl-source-code-encryption

Incisive、Xcelium 和 Genus 确实支持用于源代码加密的 IEEE1753 标准。

您需要的是使用 ncprotect 加密文件。默认保护机制使用 Cadence 专有加密算法，不需要许可证。Cadence 还支持 IEEE 加密机制。但是，要调用 IEEE 加密，需要许可证。要加密文件，您需要下载并安装 Incisive 模拟器工具集。 1) 下载并安装 Incisive 模拟器。 2) 一旦你安装了工具，使用你的 vhdl 文件执行 ncprotect 并包含 -autoprotect : ncvhdl myfile.vhd -autoprotect 以上将导致完全加密（端口、名称、层次结构等……）生成的文件与输入文件同名，但扩展名为 .vhdp。

给定加密算法和加密密钥，ncprotect 工具基本上执行文本加密。由于 ncprotect 使用标准 RSA 库，因此可以选择 RSA、RC2、RC4、AES、DSA 等各种算法。 ncprotect 工具被移植到各种 32 位和 64 位平台上，如 HP、Linux、IBM、Solaris、windows 等。因此，对于各种 IP 开发人员来说，使用它变得非常方便。

英特尔® Quartus® Prime 专业版加密 Verilog 或 VHDL 文件

英特尔® Quartus® Prime 专业版软件支持用于 IP 内核文件解密的 IEEE 1735 v1 加密标准。您可以使用 encrypt_1735 实用程序或支持 IEEE 1735 标准的第三方加密工具来加密 Verilog HDL 或 VHDL IP 文件。然后，您可以在支持 IEEE 1735 加密标准的英特尔® Quartus® Prime 专业版软件和仿真工具中使用加密文件。

Verilog HDL 和 VHDL 的加密密钥相同。您可以使用与非加密模块相同的方法将参数传递给加密模块的实例化。

只有使用第三方工具的文件加密需要公共加密密钥。使用英特尔® Quartus® Prime 专业版软件加密文件不需要公共加密密钥。

要使用英特尔® Quartus® Prime 专业版软件加密 Verilog HDL 或 VHDL 文件，请使用位于 /quartus/bin 文件夹中的命令“encrypt_1735”。

encrypt_1735 <文件名>.v --language=verilog --quartus
encrypt_1735 <文件名>.vhd --language=vhdl --quartus

注意：英特尔® Quartus® Prime 标准版软件不支持 IEEE 1735 加密。

IP 加密工具
https://ipencrypter.com/wp-content/uploads/2016/11/ipe1735v2-1610-1-0-ug01.pdf

IP 作者可以使用 IP Encrypter 工具按照 IEEE Std 1735™-2014 (IEEE P1735 v2) 标准对 IP 进行加密。 IP 作者可以通过通用和工具块中的保护指令提供保护级别。 IP 作者可以选择支持的工具，每个工具都需要公钥。

通过 ipecrypt 创建加密 IP 的过程：

- 以纯文本创建 IP
- 在代码周围添加保护指令以加密或通过单独的文件指定指令
- 运行 ipecrypt 应用程序以生成加密 IP

Siemens Mentor Graphics QuestaSim、Modelsim
Siemens Mentor Graphics QuestaSim、Modelsim

ModelSim 的加密解决方案允许 IP 作者为各种 EDA 工具和设计流程提供加密的 IP 代码。

ModelSim 通过受保护的加密信封支持 VHDL、Verilog 和 SystemVerilog IP 代码加密。 VHDL 加密由 IEEE Std 1076-2008 第 24.1 节（标题为“保护工具指令”）和附件 H，第 H.3 节（标题为“数字信封”）定义。 Verilog 加密由 IEEE Std 1364-2005 第 28 节定义； SystemVerilog 加密由 IEEE Std 1800-2012 第 34 节定义（两个部分的标题为“受保护的信封”）。数字信封使用模型是 VHDL `protect
和 Verilog 的 `pragma 保护编译器指令。

ModelSim 支持 IEEE P1735-2014 工作组关于不同加密和解密工具之间的加密互操作性建议的版本 1。它解决了使用模型、算法选择、约定和对 HDL 标准的微小修正，以实现有用的互操作性。

ModelSim 使用两种受保护的形式：
- 第一种形式是一个文本文件，其中包含输入原始纯文本 HDL 源文件的转换版本。
- 第二种形式是您编译的设计单元的受保护版本。

Verilog 和 SystemVerilog 的 ModelSim vencrypt 实用程序只生成文本文件。它不会将任何东西编译到库中，也不会处理宏或处理通常的 Verilog 开关。相比之下，Verilog/SystemVerilog 编译命令 vlog +protect 生成文本文件，将它们编译到库中，并处理宏（以及所有其他常用的 vlog 参数）。

VHDL 的 ModelSim vhencrypt 实用程序与 vencrypt 实用程序的工作方式相同（尽管 VHDL 没有宏）。 VHDL 编译命令 vcom +protect 的工作方式相同 编译源代码时，您可以使用 vcom -nodebug 或 vlog -nodebug 命令对最终用户隐藏编译结果。

Synopsys IP 加密和部署套件 VCS（Verilog 编译器和模拟器）、VMC（Verilog 模型编译器）和 VHMC（VHDL 模型编译器）

Synopsys 提供一整套工具来处理 IP 加密和部署。具体到加密，我建议您参考 VCS（Verilog 编译器和模拟器）、VMC（Verilog 模型编译器）和 VHMC（VHDL 模型编译器）产品。这些将源 RTL 转换为非常安全的加密包，然后可以安全地交付给客户。加密模型可以插入大多数商业模拟器。

可交付模型中没有原始来源的痕迹，因此无法“破解”。

Xilinx Vivado 设计套件 IP 加密

https://support.xilinx.com/s/question/0D52E00006hpVtJSAU/how-can-i-encrypt-rtl-source-file?language=en_US
https://support.xilinx.com/s/question/0D52E00006zybDsSAI/how-to-encrypt-my-ip-into-secureip-like-logicore-ip-in-xps?language=en_US

http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_4/sim.pdf
http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_2/irn.pdf

在 Vivado 2016.3 及更高版本中，赛灵思增加了使用 IEEE 1735 v2 加密对 HDL 源文件进行加密的功能。赛灵思 Vivado Design Suite® 支持符合 IEEE-1735-2014 版本 2 的加密。 IP 加密涵盖 HDL（SystemVerilog、Verilog、VHDL）设计入口直至比特流生成。 IP 作者可以通过表达工具应如何与 IP 交互来管理其 IP 的访问权限。保护比特流本身需要对已编程 Xilinx® 器件上的比特流进行加密，这一步骤通常由 IP 用户而非 IP 创建者完成。

尽管设计源文件的格式有很多种，但 IEEE-1735-2014 仅适用于 Verilog、SystemVerilog 和 VHDL 格式。这些 RTL 标准也由 IEEE 管理，并且通过相互协议，这些标准将允许 IEEE-1735 定义 IP 安全中的行为，直到 IEEE 1735 中的建议可以修改为包含在语言
参考手册 (LRM)。 IEEE 1735 不涵盖流行的网表格式，如 EDIF。

IEEE-1735 v2 加密功能需要许可证，可发送电子邮件至 xilinx_security_app@xilinx.com 索取。

注意：加密许可证功能与特定的 Vivado 版本相关联。申请许可证时，请提供需要许可证的 Vivado 的确切版本（例如，2016.3、2016.4）。请确保在申请许可证时使用公司/公司电子邮件地址。

使用 EDA 和 FPGA 保护 IP 核 – 一个建议

这是一个关于使用 EDA 和 FPGA 设计保护 IP 的建议。

首先，假设 EDA 软件是可信的。第三方设计的 IP 核必须先从 EDA 软件开发商处购买 RTL 级 IP 核。第三方用 EDA 开发者的公钥对 IP 核进行加密，然后使用设计者提供的公钥。这样，设计者虽然获得了第三方 IP 核，但不知道 EDA 软件开发者的私钥，所以设计者无法知道 IP 核的 RTL 级描述。

设计人员可以将购买的 IP 核视为具有已知功能的黑匣子。使用黑盒和其他电路模块完成电路描述后，设计人员将设计导入 EDA 工具。 EDA 工具使用设计人员输入的密钥和 EDA 开发人员的密钥对加密的 IP 核进行两次解密，以获得整个设计的完整 RTL 表示。然后，EDA 工具可以对设计进行行为综合、逻辑综合、技术映射，最后在技术映射后生成网表。假设用于综合和技术映射的工具由 EDA 开发者提供，FPGA 布局布线工具由另一个 EDA 开发者提供，例如 FPGA 制造商提供。

在将优化后的门级网表写入文件之前，EDA 工具首先用自己的私钥加密网表，然后用布局布线工具提供商的公钥加密网表。这样，设计人员在综合和技术映射后仍然无法查看网表，从而防止设计人员使用逆向工程窃取 IP 核。

布局布线工具先用自己的私钥对加密后的网表进行解密，然后使用前端 EDA 工具开发者的公钥进行二次解密，得到解密后的网表。使用这个网表，layout 工具可以完成设计在芯片上的 layout，最终生成一个用于 FPGA 配置的 bit-stream 文件。

笔记：

本文是“第 17 部分 IP 核保护”一书章节的一部分，该章节也是由 Mark Chen 编写的“硅 IP – 不仅仅是设计”一书的一部分。您还可以在以下网站 上找到更多信息。