CASignTool

使用说明

文档版本　04

发布日期　2015-05-25

商标声明

 、HISILICON、海思和其他海思商标均为深圳市海思半导体有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市海思半导体有限公司

地址：　深圳市龙岗区坂田华为基地华为总部　邮编：518129

网址：　http://www.hisilicon.com

客户服务邮箱：　support@hisilicon.com

前 言

概述

本文档主要介绍如何通过 CASignTool 工具生成高级安全系列芯片专用 boot。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3716C 芯片	V100/V200
Hi3716H 芯片	V100
Hi3716M 芯片	V200
Hi3716M 芯片	V300/V400/V310/V420/V410
Hi3110E 芯片	V200/V300/V400/V500
Hi3719C 芯片	V100
Hi3719M 芯片	V100
Hi3796C 芯片	V100
Hi3798C 芯片	V100

注意：Hi3110E V200/V300/V400、Hi3716CV100、Hi3716HV100、Hi3716MV200 芯片签名方案一致，Hi3716MV300/V400/V310/V420/V410、Hi3110EV500、Hi3716CV200、Hi3719CV100、Hi3719MV100、Hi3796CV100、Hi3798CV100 芯片高安方案一致。

读者对象

本文档（本指南）主要适用于以下工程师：

- 技术支持工程师

- 软件开发工程师

作者信息

章节号	章节名称	作者信息
全文	全文	L00168554

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

修订日期	版本	修订说明
2012-03-27	00B01	第 1 次临时版本发布。
2012-05-10	00B02	增加 MV300 的使用说明。
2012-12-07	00B03	更新 MergeImage 的使用方法。
2013-03-20	00B04	按照特定 CA 公司和 STB 厂商要求更新 MergeImage 的使用方法。
2013-08-19	00B05	更新签名工具芯片类型选择。
2013-09-16	00B06	增加产生 RSA 密钥功能，加解密镜像功能以及验证签名菜单；修改工具子菜单名称；支持 Hi3716CV200 / Hi3719CV100 / Hi3719MV100。
2014-04-02	00B07	新增 2.1.3.1、2.1.3.2 章节；修改 2.5 章节。
2014-04-15	00B08	全文增加支持 Hi3716MV400 高安芯片的说明。
2014-04-18	00B09	修改单板配置参数的使用方式。
2014-06-05	01	支持 Hi3796CV100 / Hi3798CV100。更新 2.6 章节。添加 2.8 章节说明
2014-10-21	02	支持 Hi3716MV310 芯片，增加 boot 加密支持。
2014-12-10	03	Merge Signed-BootImage 页面增加 boot 加密支持。
2015-05-25	04	新增支持 Hi3716MV420/Hi3716MV410 和 Hi3110EV500。

目 录

插图目录

表格目录

1 工具介绍

1.1 概述

高级安全系列芯片具有安全启动功能，即启动时芯片自动检查 boot 的签名。高级安全系列芯片不能直接使用普通的 boot（SDK 开发包中直接生成的，普通芯片用的 boot），而需要高级安全系列芯片的专用 boot（本文中称这样的 boot 为安全 boot）。CASignTool 用于将开发包中生成的普通 boot 打包生成安全 boot，同时也支持将 Kernel、文件系统等合成 Image 文件，方便机顶盒厂商提交镜像给 CA 厂商签名。

高级安全系列芯片的安全启动使用的签名校验算法是：RSA2048 bits PKCS #1 v1.5 with SHA256。

1.2 功能描述

1.2.1 功能简介

安全启动功能的大致实现流程为：

步骤 1 开发调试 BOOT。

高级安全系列芯片出厂时写入了对应 CA 厂商的签名密钥，但安全启动功能未开启。这时芯片不对 BOOT 进行签名检查，开发人员拿这样的芯片开发调试安全 BOOT。

步骤 2 申请签名。

开发人员将 BOOT 开发完成后，向 CA 厂商申请签名，并开启安全启动功能。产品（如机顶盒）生产时，必须要写入带签名的 BOOT，并开启安全启动功能。

----结束

从芯片对 BOOT 的要求来划分，海思芯片可分为 3 类：

- 普通芯片。直接使用 SDK 包中生成的普通 BOOT。
- 高级安全系列芯片，安全启动未使能。使用高级安全芯片专用 BOOT，不用签名。
- 高级安全系列芯片，安全启动使能。使用高级安全芯片专用 BOOT，必须签名。

表 1-1 列出了 3 种不同 BOOT 的适用范围。

表1-1 不同 BOOT 的适用范围

BOOT 类型	普通芯片	高级安全芯片，安全启动未使能	高级安全芯片，安全启动使能
普通 BOOT	√	×	×
安全 BOOT，不带签名	×	√	×
安全 BOOT，带签名	×	√	√

CASignTool 支持以下功能：

- 生成不带签名的安全 BOOT
- 生成带签名的安全 BOOT
- 生成自签名的安全 BOOT
- 合成高安芯片启动所需要的 BootImage 文件，海思将该文件命名为 FinalBoot.bin
- 提供应用程序，包括 Kernel, FS 等 Image 文件的签名工具
- 提供机顶盒厂商测试所需要的 RSA Key 生成工具
- 提供可用于验证签名 BOOT 正确性的工具(仅模拟验证使用)
- 提供 AES,TDES 等加解密工具
- Nagra 模拟签名工具

2 操作步骤

2.1 生成不带签名的安全 BOOT 功能

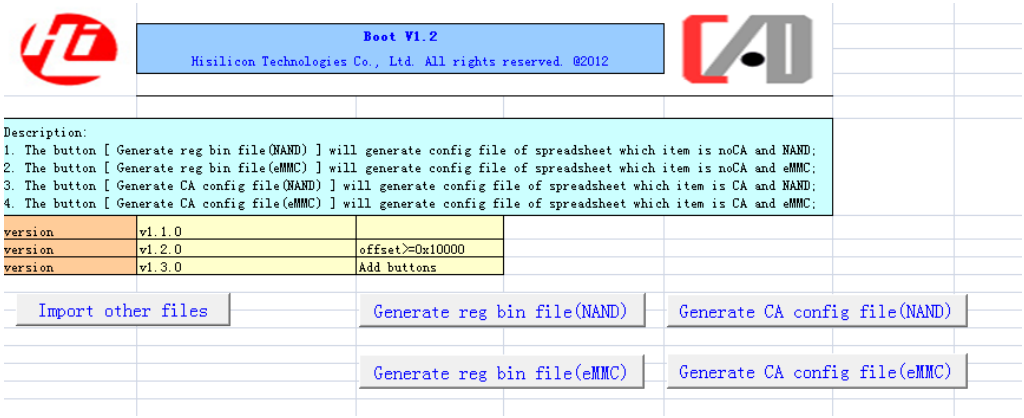
该功能生成的 BOOT 不带签名，适用于安全启动未使能的高级安全芯片。多用于开发人员开发安全 BOOT 时使用。

2.1.1 环境准备

在生成安全 BOOT 之前，需要做如下准备：

步骤 1 单板配置参数。以 Hi3716CV200 芯片为例。BOOT 参数跟单板相关，开发人员需要使用海思 SDK 中提供的 BOOT 配置 excel 工具来配置单板的参数，然后用这个工具生成一个*.cfg 文件。如图 2-1 所示。

图2-1 BOOT 配置 excel 工具界面



表格配置好后，根据单板所使用的 flash 类型单击 Generate CA config file(NAND) 或 Generate CA config file(eMMC) 的按钮，工具会生成对应的*.cfg 文件。如使用 nand flash 的 hi3716cdmo2b_hi3716cv200_ddr3_2gbyte_8bitx4_4layers.xlsm 表格所产生的配置参数文件名称为：hi3716cdmo2b_hi3716cv200_ddr3_2gbyte_8bitx4_4layers_nand.cfg。Hi3716CV100/Hi3716HV100/Hi3716MV200 对应的表格生成的*.cfg 文件大小不能超过

0x3d0; Hi3716MV300、后续芯片对应的表格生成的*.cfg 文件大小不能操过 0x13d0。如果超过了，请联系海思进行缩减参数配置。这个 excel 工具中 Generate reg bin file(NAND) 和 Generate reg bin file(eMMC) 的按钮是用来生成普通 BOOT 的配置参数的，请注意不要混淆。

Hi3716CV200 以及后续芯片，*.cfg 的尾部数据添加了产生该文件的时间和原文件注释信息。客户可以使用 UltraEdit 等工具查看*.cfg 的信息。

步骤 2 使用 SDK 编译生成普通 BOOT。

----结束

说明

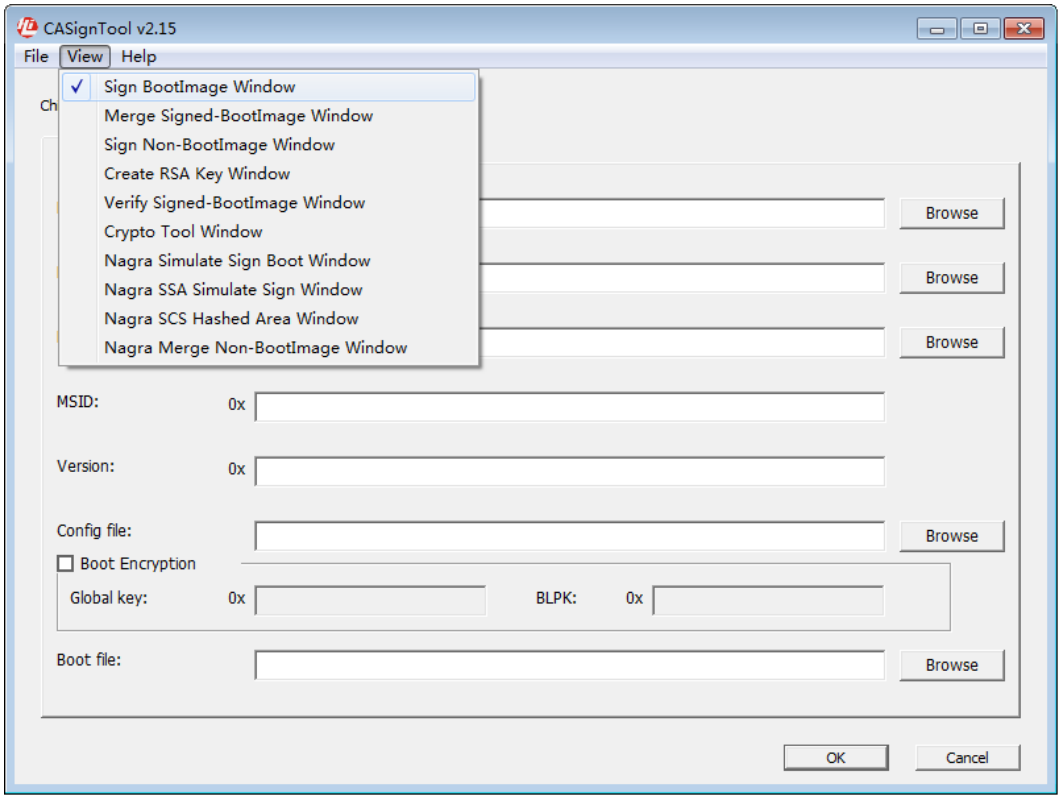
本文后续所指的 cfg.bin 文件，就是根据 BOOT 配置 excel 工具所生成的*.cfg 文件。

2.1.2 操作流程

操作步骤如下：

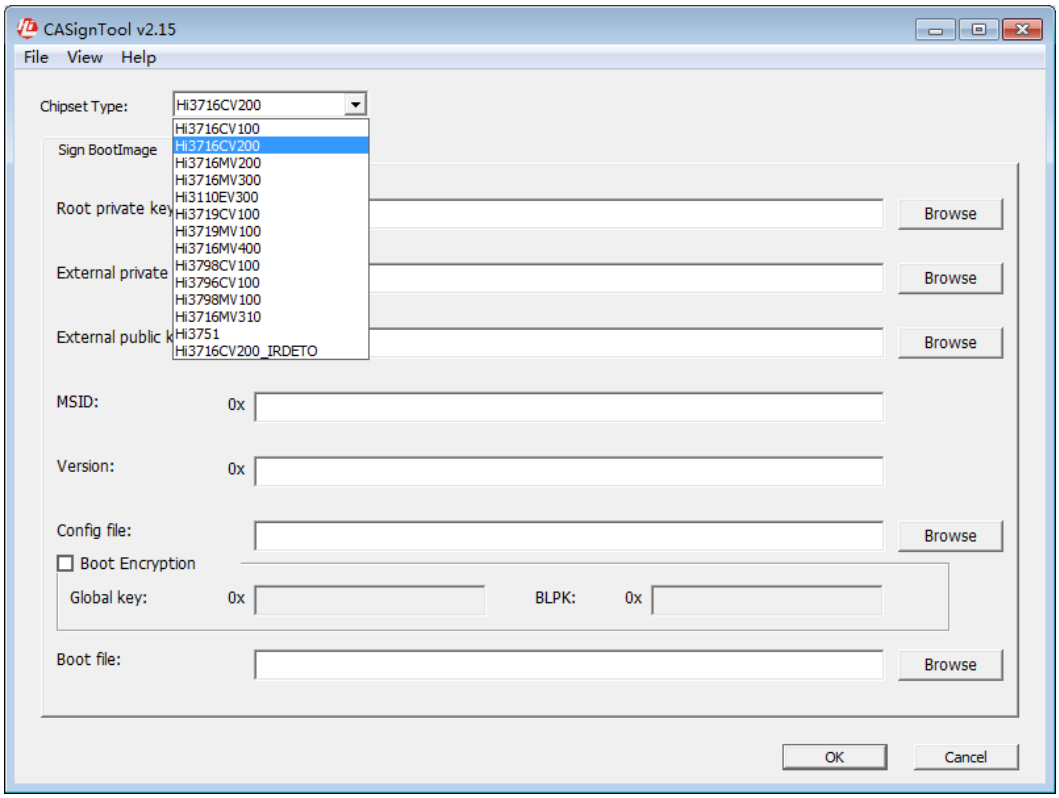
步骤 1 打开 CASignTool 工具，选择“Sign BootImage”页，如图 2-2 所示。

图2-2 CASignTool 工具“Sign BootImage”页界面



步骤 2 选择芯片类型。

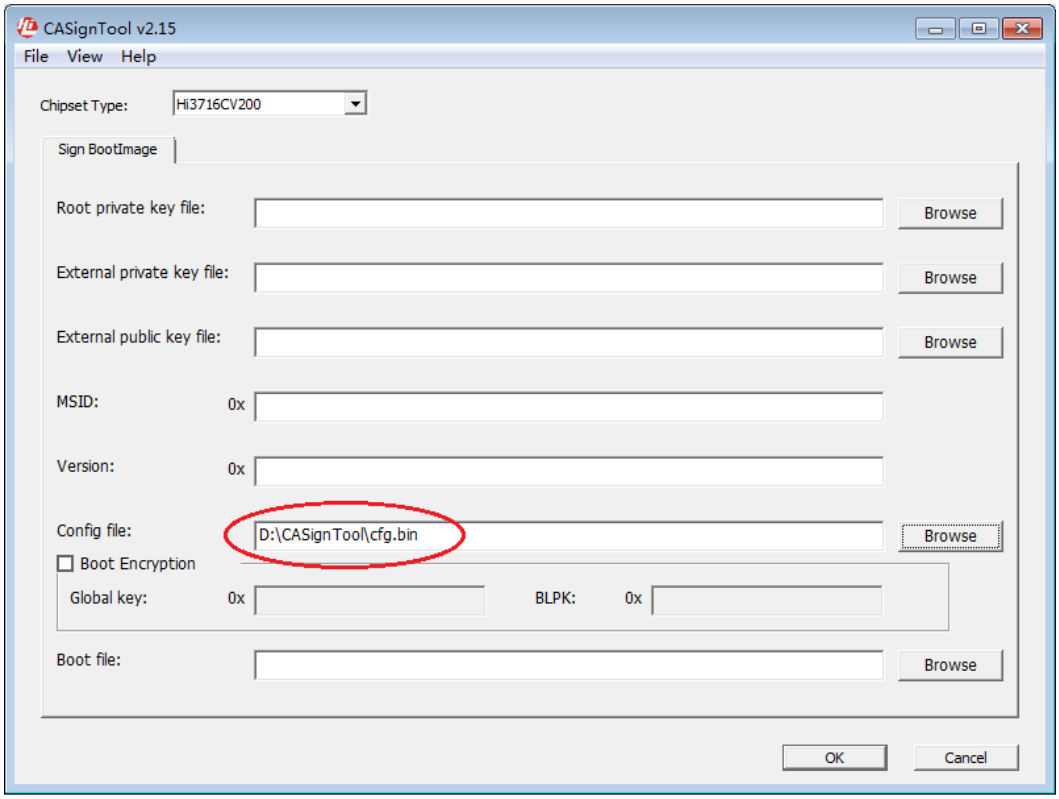
图2-3 选择芯片类型



当前支持的芯片型号包括：Hi3716CV100，Hi3716CV200，Hi3716MV200，Hi3716MV300，Hi310EV300，Hi310EV500，Hi3719CV100，Hi3719MV100，Hi3716MV400，Hi3796CV100，Hi3798CV100，Hi3798MV100，Hi3716MV310，Hi3716MV410，Hi3716MV420 等高安芯片。请选择正确的芯片型号。

步骤 3 选择 2.1.1 步骤 1 中准备的 cfg.bin。

图2-4 选择 cfg.bin

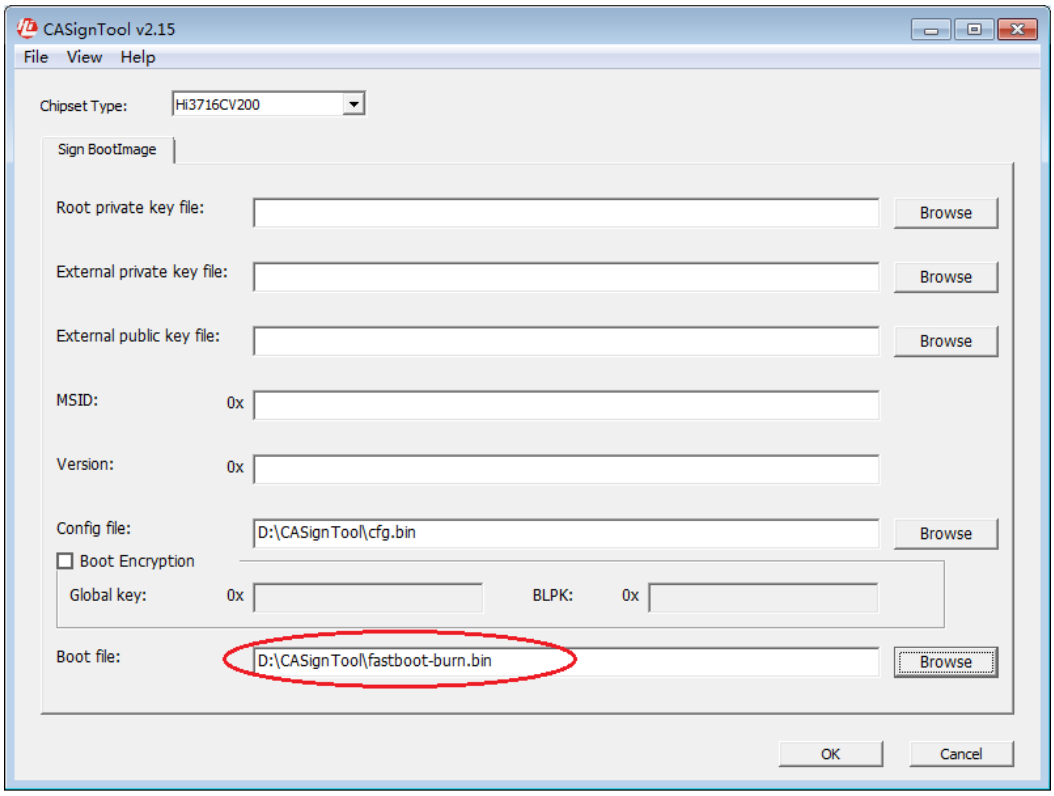


步骤 4 选择 2.1.1 步骤 2 中准备的 BOOT。

该 BOOT 文件需要在 SDK 发布包中将高安编译选项打开后编译产生。

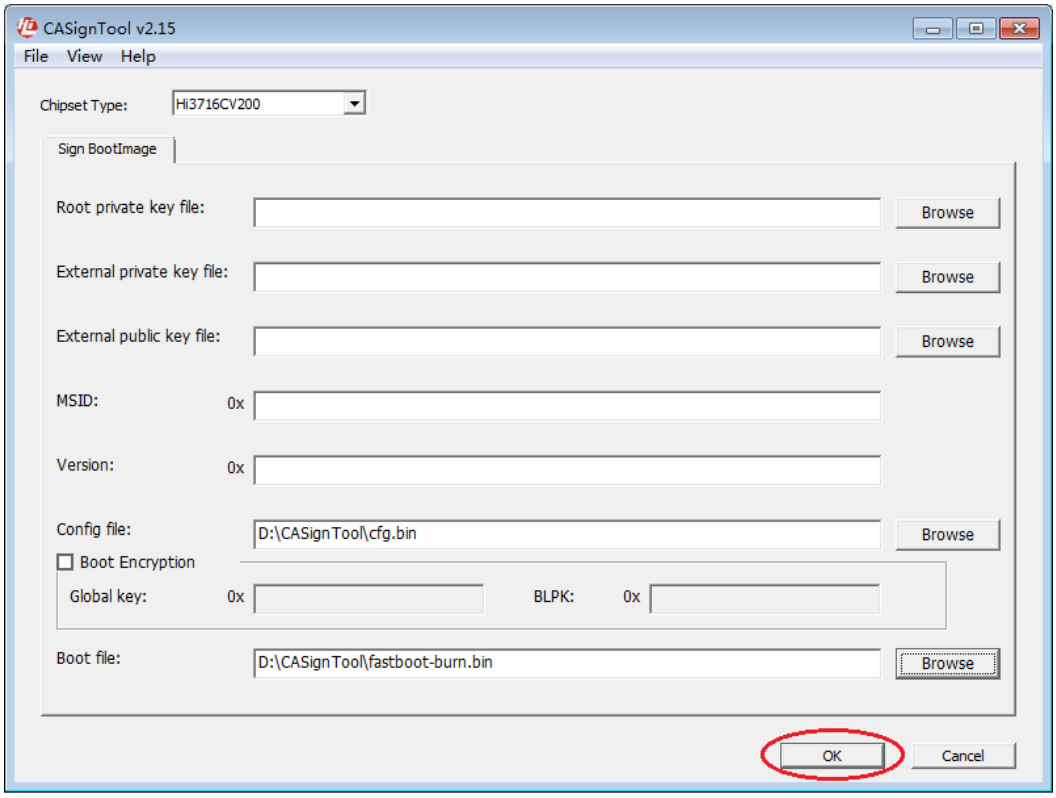
- 在高清 SDK 包中产生的 BOOT 为 fastboot_burn.bin.
- 在标清 SDK 包中产生的 BOOT 为 miniboot2008.bin.

图2-5 选择 BOOT



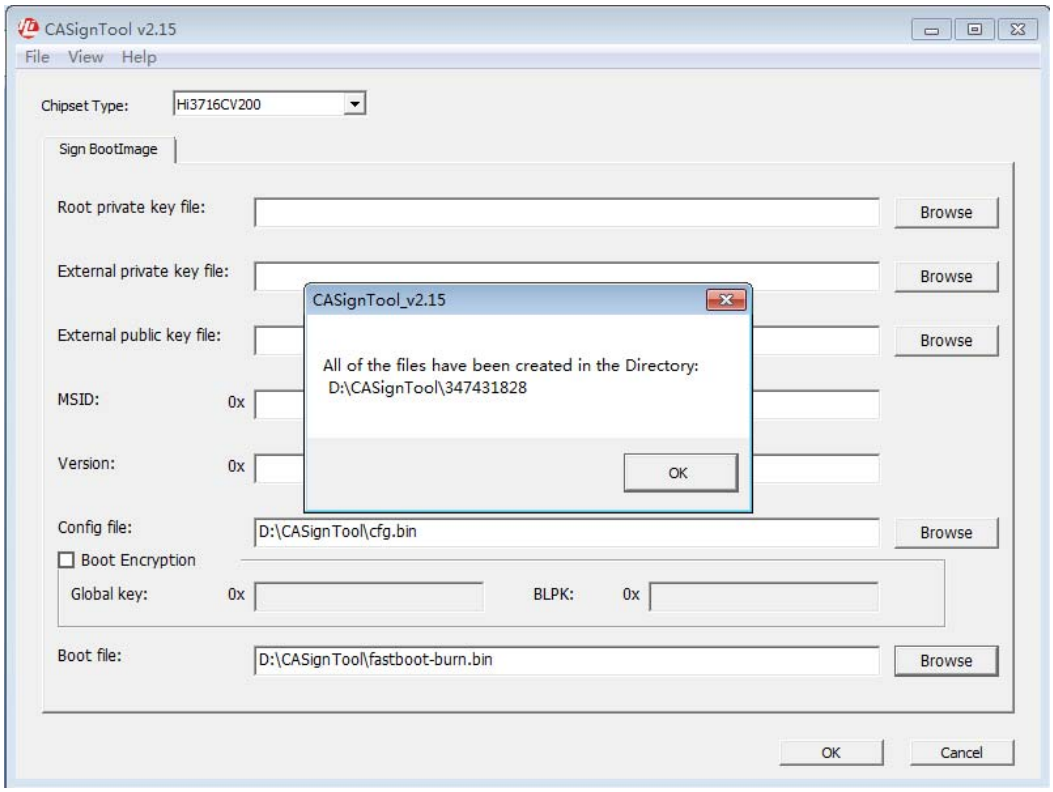
步骤 5 单击“OK”键。

图2-6 单击“OK”键



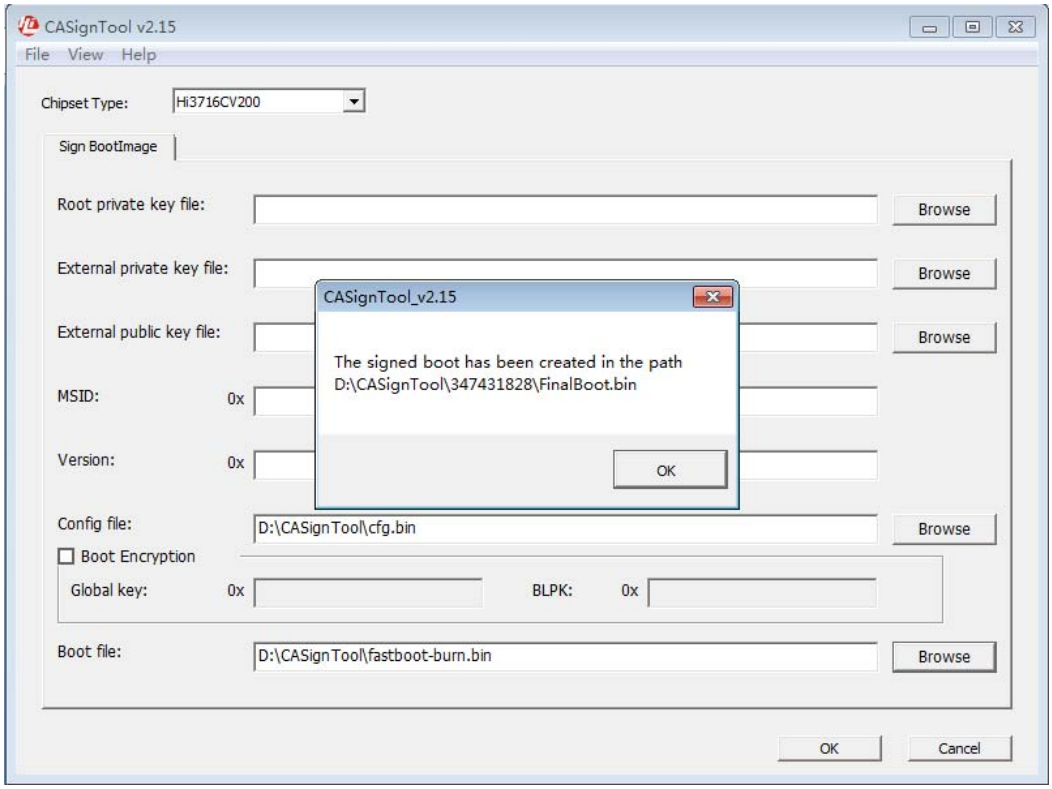
步骤 6 工具将会在工具所在的文件夹新建一个文件夹，生成文件放在这个文件夹下。示例中 347431828 这个文件夹名为随机生成，与实际生成会不一致，请以实际提示为准。

图2-7 文件生成提示



步骤 7 生成不带签名的安全 BOOT 的名字为：FinalBoot.bin。

图2-8 BOOT 成功生成提示



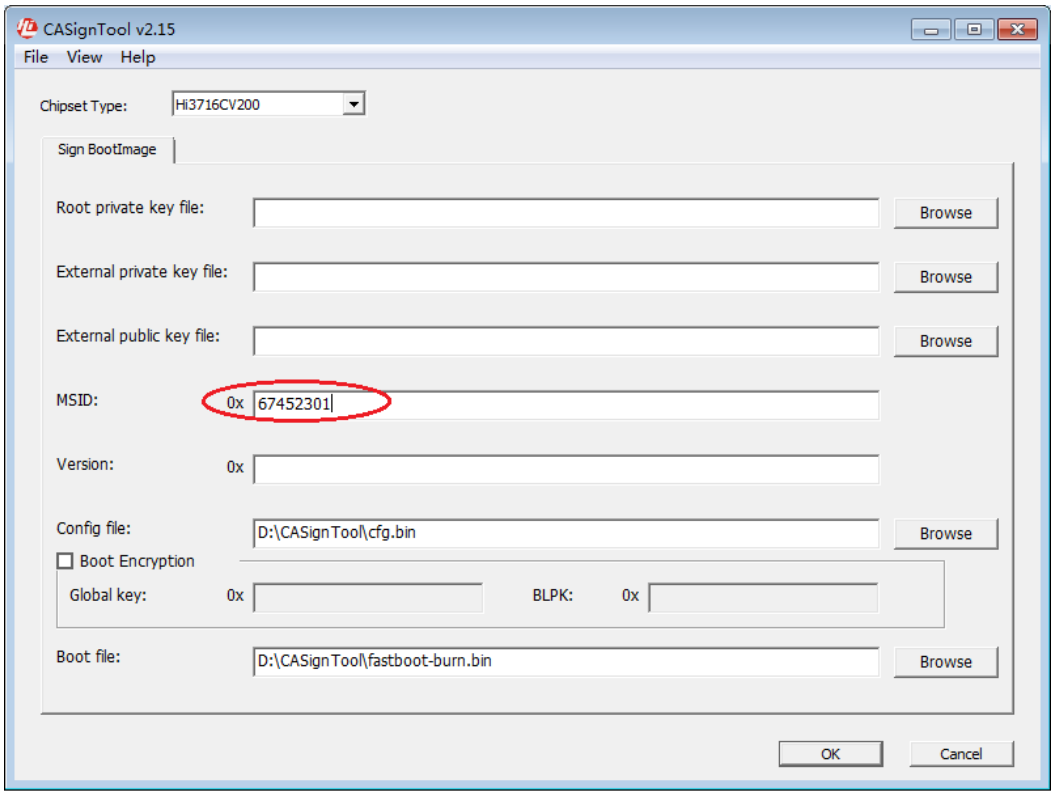
----结束

2.1.3 注意事项

2.1.3.1 设置市场区域码

- 如果需要配置市场区域码，请在“2.1.2 操作流程”的“2.1.2 步骤 1”操作过程中，选择“Sign BootImage”页，并填入市场区域码。其他操作与“2.1.2 操作流程”的 2.1.2 步骤 2~2.1.2 步骤 7 一样。

图2-9 配置市场区域码的“Sign BootImage”页界面



- 如果 CA 厂商要求设置市场区域码(MSID)，现在以 MarketID=0x67452301 为例，注意如下的使用规则。

对于非 NAGRA CA 芯片

表2-1 非 NAGRA CA 芯片

Item	Hi3716CV200/Hi3719CV100/Hi3719MV100/Hi3716MV400/Hi3798MV100/Hi3716MV420/Hi3716MV410/Hi3110EV500	Hi3716MV300/Hi3716MV310	Hi3716CV100/Hi3716MV200	Hi3110E
HI_UNF_ADVCA_SetMarketId(HI_U8 u8MarketId[4])	u8MarketId[0] = 0x67 u8MarketId[1] = 0x45 u8MarketId[2] = 0x23 u8MarketId[3] = 0x01	u8MarketId[0] = 0x67 u8MarketId[1] = 0x45 u8MarketId[2] = 0x23 u8MarketId[3] = 0x01	u8MarketId[0] = 0x01 u8MarketId[1] = 0x23 u8MarketId[2] = 0x45 u8MarketId[3] = 0x67	u8MarketId[0] = 0x01 u8MarketId[1] = 0x23 u8MarketId[2] = 0x45 u8MarketId[3] = 0x67

Item	Hi3716CV200/Hi3719CV100/Hi3719MV100/Hi3716MV400/Hi3798MV100/Hi3716MV420/Hi3716MV410/Hi3110EV500	Hi3716MV300/Hi3716MV310	Hi3716CV100/Hi3716MV200	Hi3110E
CASignTool	0x67452301	0x67452301	0x67452301	0x67452301
System Register	0xf8ab0120：0x01 0xf8ab0121：0x23 0xf8ab0122：0x45 0xf8ab0123：0x67	0x10180120：0x01 0x10180121：0x23 0x10180122：0x45 0x10180123：0x67	0x10000008：0x01 0x10000009：0x23 0x1000000a：0x45 0x1000000b：0x67	0x101c0008：0x01 0x101c0009：0x23 0x101c000a：0x45 0x101c000b：0x67
OTP (Internal use)	0xa0：0x01 0xa1：0x23 0xa2：0x45 0xa3：0x67	0xa0：0x01 0xa1：0x23 0xa2：0x45 0xa3：0x67	0x2c：0x01 0x2d：0x23 0x2e：0x45 0x2f：0x67	0x2c：0x01 0x2d：0x23 0x2e：0x45 0x2f：0x67

对于 NAGRA CA 芯片

表2-2 NAGRA CA 芯片

Item	Hi3716CV200/Hi3719CV100/Hi3719MV100/Hi3716MV400/Hi3798MV100/Hi3716MV420/Hi3716MV410/Hi3110EV500	Hi3716MV300/Hi3716MV310	Hi3716CV100/Hi3716MV200	Hi3110E
csdSetMarketSegmentId(const TCsd4BytesVector xMSId)	xMSId [0] = 0x67, xMSId [1] = 0x45, xMSId [2] = 0x23, xMSId [3] = 0x01	xMSId [0] = 0x67, xMSId [1] = 0x45, xMSId [2] = 0x23, xMSId [3] = 0x01	xMSId [0] = 0x67, xMSId [1] = 0x45, xMSId [2] = 0x23, xMSId [3] = 0x01	xMSId [0] = 0x67, xMSId [1] = 0x45, xMSId [2] = 0x23, xMSId [3] = 0x01
CASignTool	0x67452301	0x67452301	0x67452301	0x67452301
System Register	0xf8ab0120：0x01 0xf8ab0121：0x23 0xf8ab0122：0x45 0xf8ab0123：0x67	0x10180120：0x01 0x10180121：0x23 0x10180122：0x45 0x10180123：0x67	0x10000008：0x01 0x10000009：0x23 0x1000000a：0x45 0x1000000b：0x67	0x101c0008：0x01 0x101c0009：0x23 0x101c000a：0x45 0x101c000b：0x67

Item	Hi3716CV200/Hi3719CV100/Hi3719MV100/Hi3716MV400/Hi3798MV100/Hi3716MV420/Hi3716MV410/Hi3110EV500	Hi3716MV300/Hi3716MV310	Hi3716CV100/Hi3716MV200	Hi3110E
OTP (Internal use)	0xa0：0x01 0xa1：0x23 0xa2：0x45 0xa3：0x67	0xa0：0x01 0xa1：0x23 0xa2：0x45 0xa3：0x67	0x2c：0x01 0x2d：0x23 0x2e：0x45 0x2f：0x67	0x2c：0x01 0x2d：0x23 0x2e：0x45 0x2f：0x67

以 Hi3110E 芯片为例，提供更加直观的参看方法。如图 2-10 所示。

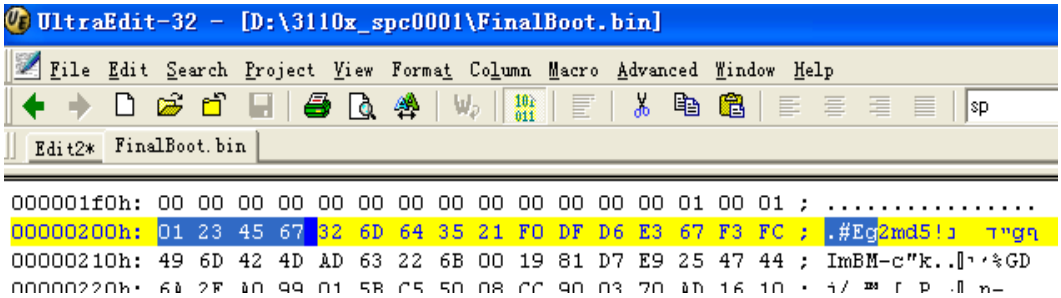
图2-10 Hi3110E 芯片市场区域码（打印信息）

```
MarketID Example:(chipset 3110E X5V400)
CASignTool.exe:
MSID Input:0x67452301,
UltraEdit-32.exe to open FinalBoot.bin: Offset:0x200: 01 23 45 67
OTP:67452301, BYTE:
  0X2C-->0x01, 0X2D-->0x23, 0X2E-->0x45, 0X2F-->0x67

hisilicon # md 60c00200
60c00200: 67452301 35646d32 d6dff021 fcf367e3

# himd.l 0x101c0000
*** Board tools : ver0.0.1_20120501  ***
[debug]: (source/utils/cmdshell.c:166)cmdstr:himd.l
====dump memory 0X101C0000====
0000:   00000432 0000000f 67452301 2a0a0a0a
0010:   67452301 00000000 00000000 00000000
```

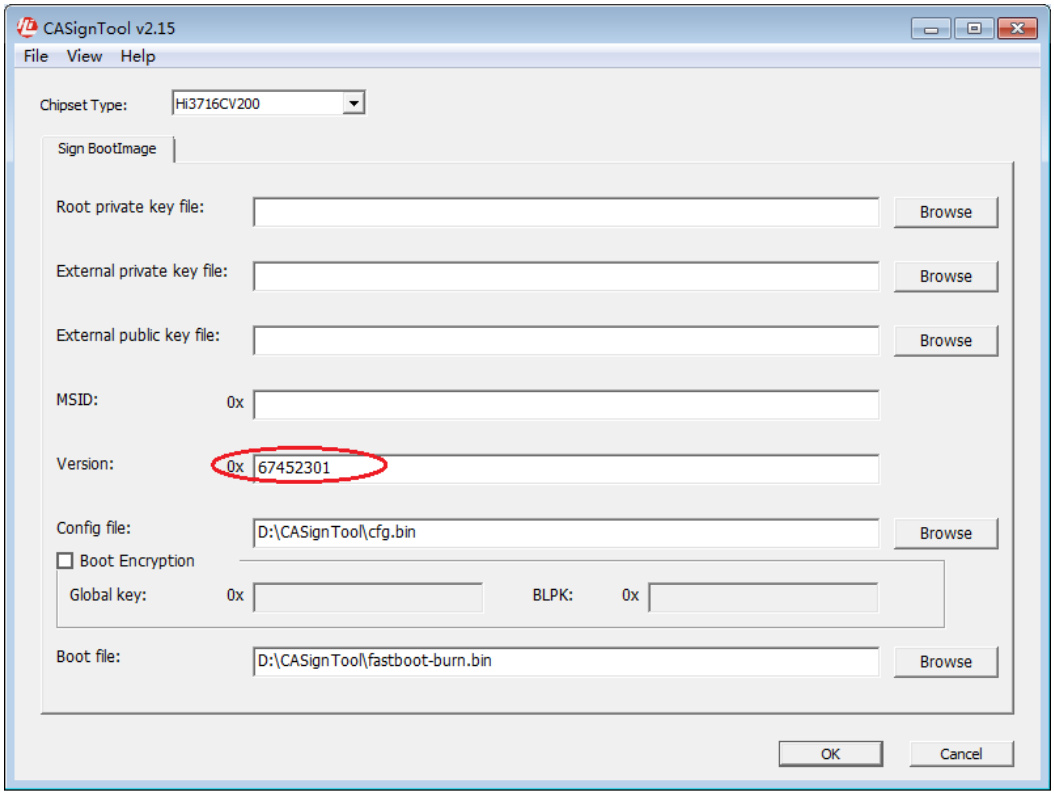
图2-11 Hi3110E 芯片市场区域码（bin 文件中）



2.1.3.2 设置 VersionID

- 如果需要配置 VersionID，请在“2.1.2 操作流程”的“2.1.2 步骤 1”操作过程中，选择“Sign BootImage”页，并填入 Version。其他操作与“2.1.2 操作流程”的 2.1.2 步骤 2~2.1.2 步骤 7 一样。
- Hi3716CV100/Hi3716MV200/Hi3110EV300/Hi3110EV400 高安芯片不支持 VersionID 的配置。
- Hi3716MV300/Hi3716CV200/Hi3719CV100/Hi3719MV100/Hi3716MV400/Hi3798MV100/Hi3716MV310/Hi3716MV420/Hi3716MV410/Hi3110EV500 等后续高安芯片支持 VersionID 的配置。
- 下图为配置 Version 的“Sign BootImage”页界面。

图2-12 设置 VersionID



- 如果 CA 厂商要求设置 VersionID，现在以 VersionID=0x67452301 为例，注意如下的使用规则：

表2-3 设置 VersionID

Item	Hi3716CV200/Hi3719CV100/Hi3719MV100/Hi3716MV400/Hi3798MV100/Hi3716MV420/Hi3716MV410/Hi3110EV500	Hi3716MV300/Hi3716MV310

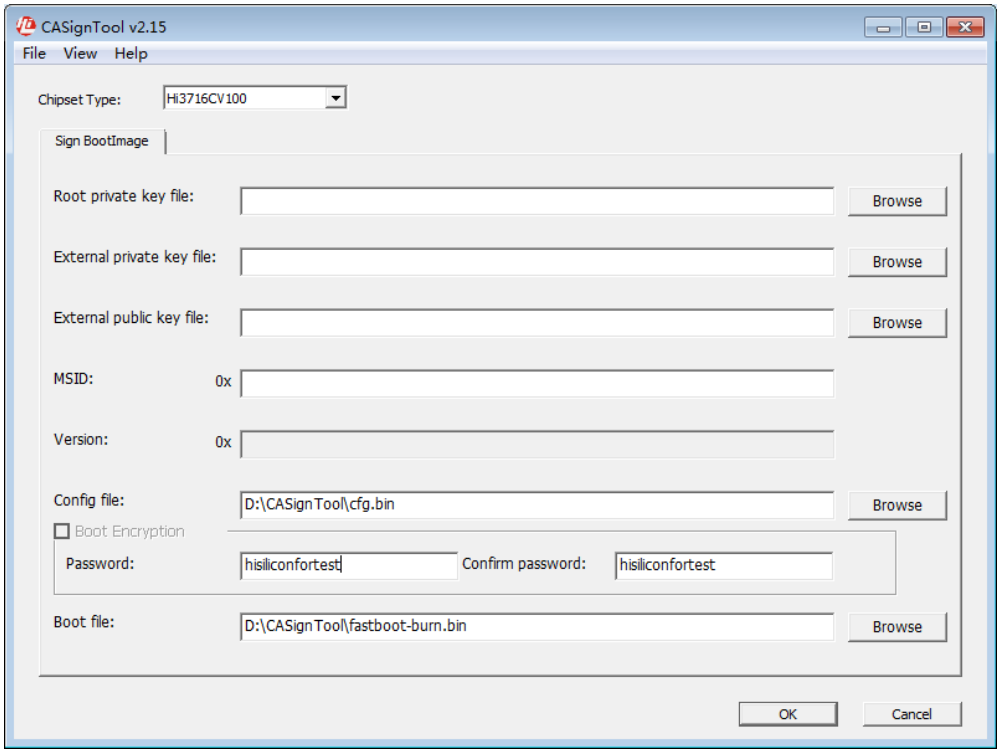
HI_UNF_ADVCA_SetVersionId(HI_U8 u8VersionId[4])	u8VersionId[0] = 0x67;
	u8VersionId [1] = 0x45;
	u8VersionId [2] = 0x23;
	u8VersionId [3] = 0x01;
		u8VersionId [0] = 0x67;
		u8VersionId [1] = 0x45;
		u8VersionId [2] = 0x23;
		u8VersionId [3] = 0x01;

CASignTool	0x67452301	0x67452301
System Register	0xf8ab0124 : 0x01; 0xf8ab0125: 0x23; 0xf8ab0126 : 0x45; 0xf8ab0127 : 0x67	0x10180124 : 0x01; 0x10180125: 0x23; 0x10180126 : 0x45; 0x10180127 : 0x67
OTP (Internal use)	0xa4 : 0x01; 0xa5 : 0x23; 0xa6 : 0x45; 0xa7 : 0x67	0xa4 : 0x01; 0xa5 : 0x23; 0xa6 : 0x45; 0xa7 : 0x67

2.1.3.3 设置 BOOT 加密

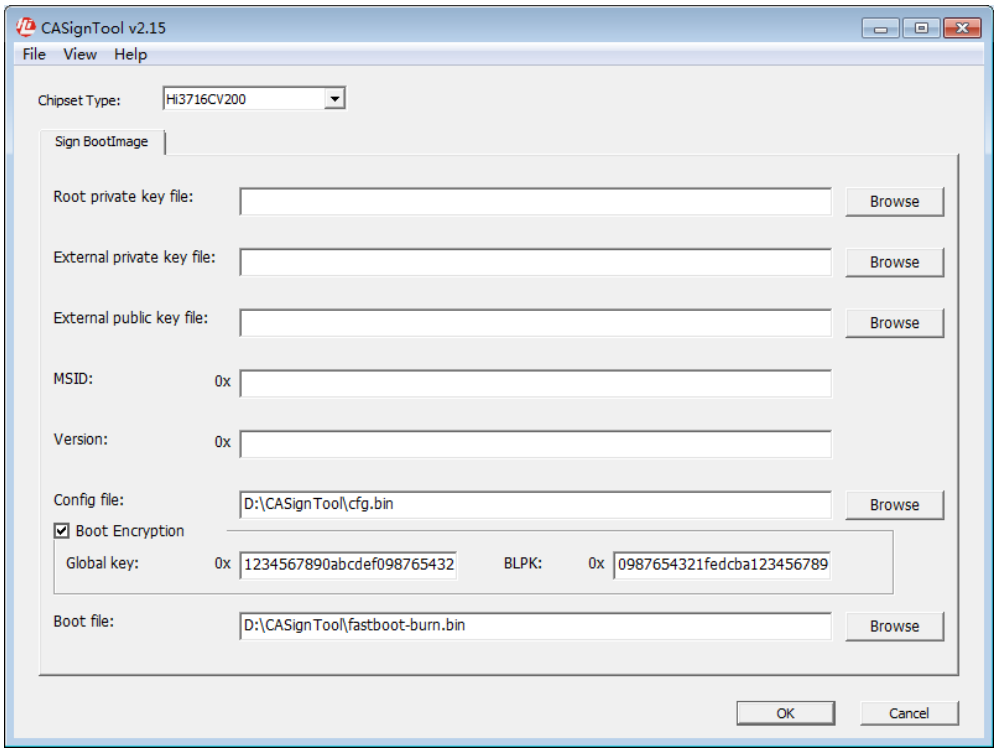
- Hi3716MV200/Hi3716CV100/Hi3110EV300 高安芯片的 BOOT 签名是采用先加密再计算数字签名的方法。这类芯片可以使用 CASignTool 对 BOOT 进行加密。如果需要加密 BOOT，请在“2.1.2 步骤 1”操作过程中，选择“Sign BootImage”页，并填入 Password，密钥的格式为 16 个 ASCII 码。其他操作与“2.1.2 操作流程”的 2.1.2 步骤 2~步骤 7 一样。

图2-13 Hi3716MV200/Hi3716CV100/Hi3110EV300 设置加密 BOOT 密钥



- Hi3716MV300/Hi3716CV200/Hi3719CV100/Hi3719MV100/Hi3716MV400/Hi3716MV310/Hi3716MV420/Hi3716MV410/Hi3110EV500 高安芯片的 BOOT 签名是采用先计算数字签名再加密的方法。对于一般的 CA 厂家来说，这类芯片不能使用 CASignTool 对 BOOT 进行加密，其加密 BOOT 的方法是先将最终 FinalBoot 写入机顶盒的 Flash 内，在机顶盒正常启动后，应用程序再执行 BOOT 加密。此时，不能勾选“Boot Encryption”选项；但对于某些特殊 CA，如 Verimatrix advanced、Irdeto 等，BOOT 加密需要用到 stb rootkey，而 stb rootkey 对于每一款芯片来说都是相同的，此时，需要勾选“Boot Encryption”选项，并填写“Global key”为 stb rootkey，“BLPK”为加密 BOOT 的保护密钥。其他操作与“2.1.2 操作流程”的 2.1.2 步骤 2~步骤 7 一样。

图2-14 Verimatrix advanced、Irdeto 设置加密 BOOT 密钥



说明

Global key 和 BLPK 都是 16 个字节的十六进制数值，各个字节之间可以由空格隔开，也可以连续填写。

2.2 生成带签名的安全 BOOT 功能

该功能生成带签名的 BOOT，适用于安全启动使能后的高级安全芯片。

2.2.1 环境准备

请参考“2.1.1 环境准备”。

2.2.2 操作流程

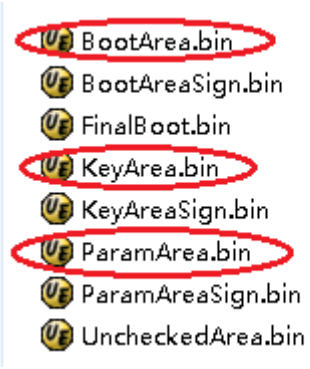
操作步骤如下：

步骤 1 按照“2.1.2 操作流程”的 2.1.1 步骤 1~2.1.2 步骤 7 操作。

步骤 2 在 STB 厂商在开发阶段，有些 CA 公司会就已经将 Ext_RSA_Pub_Key 发给 STB 厂商。在这种情况下，在使用 CASginTool 时，STB 厂商是需要在”Extern Public Key files”选择框内选择 external_rsa_ pub.txt，由此产生 KeyArea.bin。

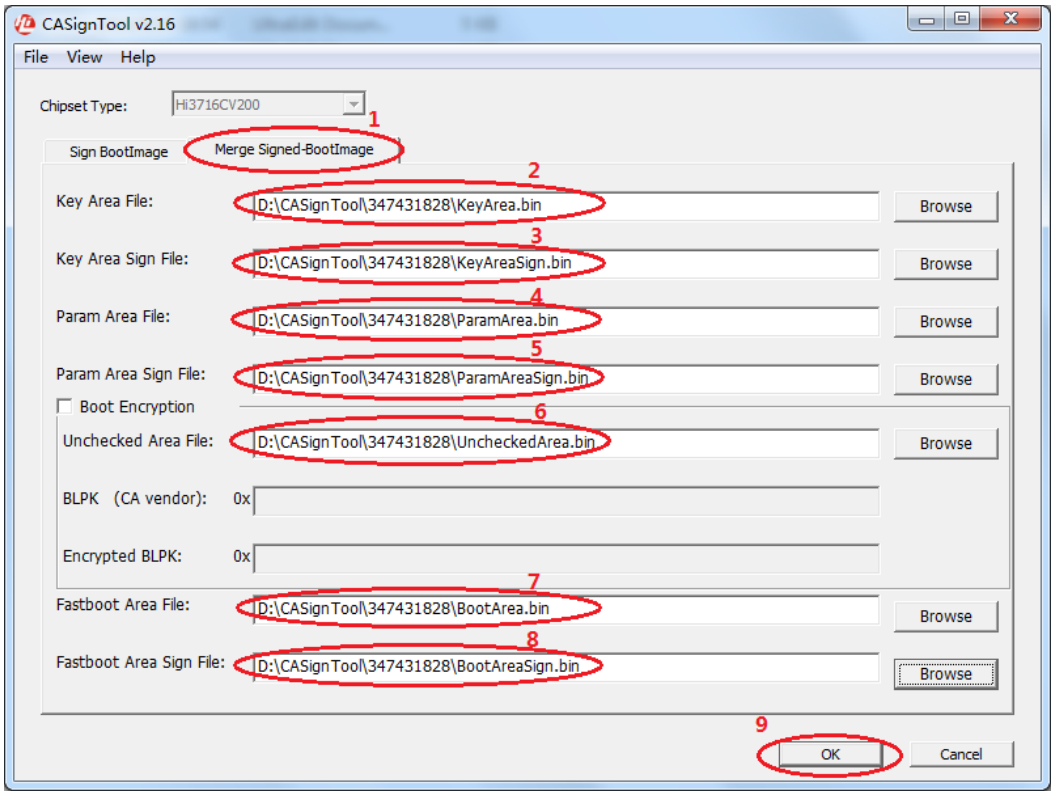
步骤 3 在生成的文件夹下，将 KeyArea.bin、ParamArea.bin 和 BootArea.bin 三个文件选出来，按照 CA 厂商的要求发给 CA 厂商申请签名。

图2-15 选择文件



步骤 4 从 CA 厂商拿到签名后，用 CASignTool 的“Merge Signed-BootImage”页合成最终文件。

图2-16 合成最终文件操作



步骤 5 Key Area File 选择第 1 步生成的 KeyArea.bin；Key Area Sign File 选择从 CA 厂商拿到的对 KeyArea.bin 的签名。

步骤 6 Param Area File 选择第 1 步生成的 ParamArea.bin 文件；Param Area Sign File 选择从 CA 厂商拿到的对 ParamArea.bin 的签名。

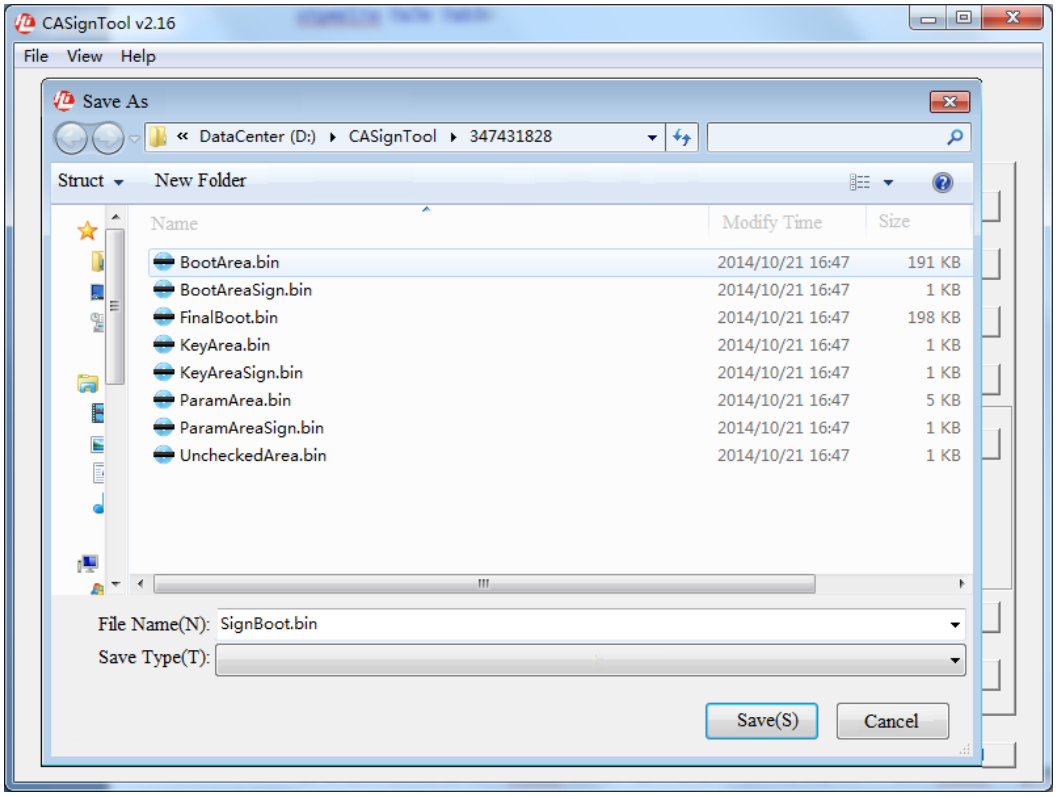
步骤 7 Unchecked Area File 选择第 1 步生成的 UncheckedArea.bin。

步骤 8 Fastboot Area File 选择第 1 步生成的 BootArea.bin 文件；Fastboot Area Sign File 选择从 CA 厂商拿到的对 BootArea.bin 的签名。

步骤 9 单击 OK。

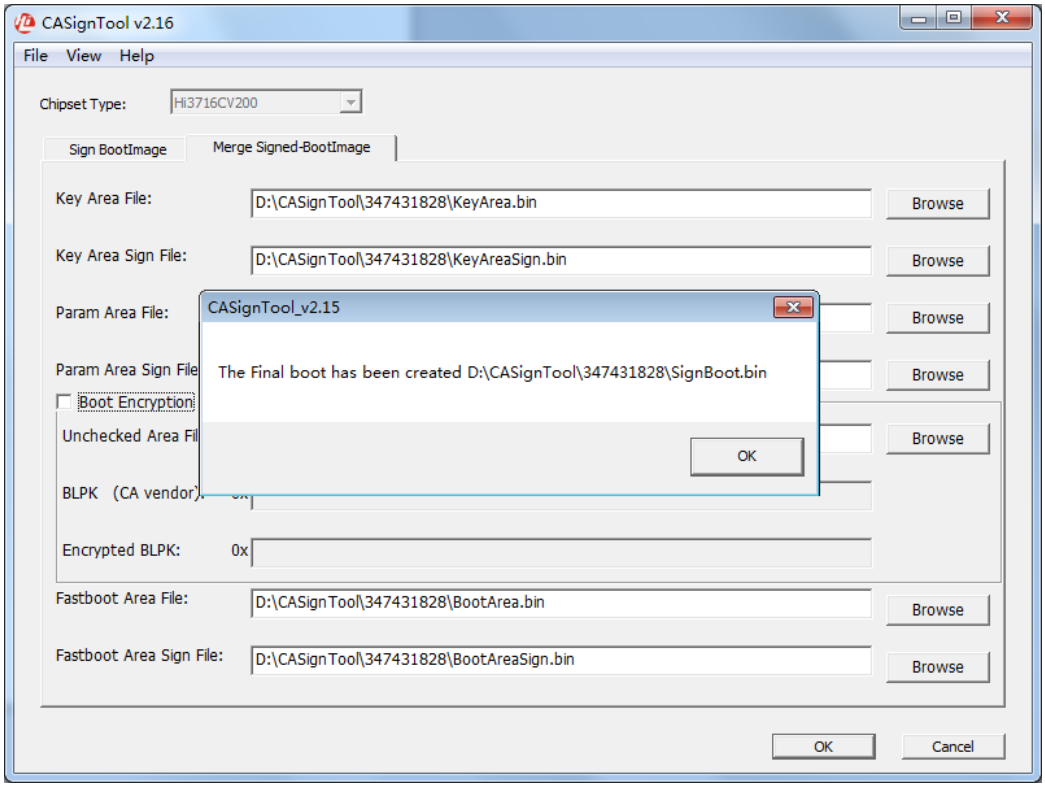
步骤 10 填写最终签名 BOOT 的名字。

图2-17 填写最终签名 BOOT 的名字



步骤 11 生成的最终签名 BOOT。

图2-18 生成的最终签名 BOOT



----结束

2.3 生成自签名的安全 BOOT 功能

当签名密钥不是由 CA 厂商掌握时，CASignTool 也支持对安全 BOOT 的签名。

2.3.1 环境准备

环境准备步骤如下：

步骤 1 配置密钥对。

海思的高级安全系列芯片对 BOOT 签名采用了 RSA2048 的密码，需要 2 个 RSA2048 密钥对：

- Root RSA 密钥对
- External RSA 密钥对

每个密钥对又分为公钥和私钥，因此总共有四个密钥：

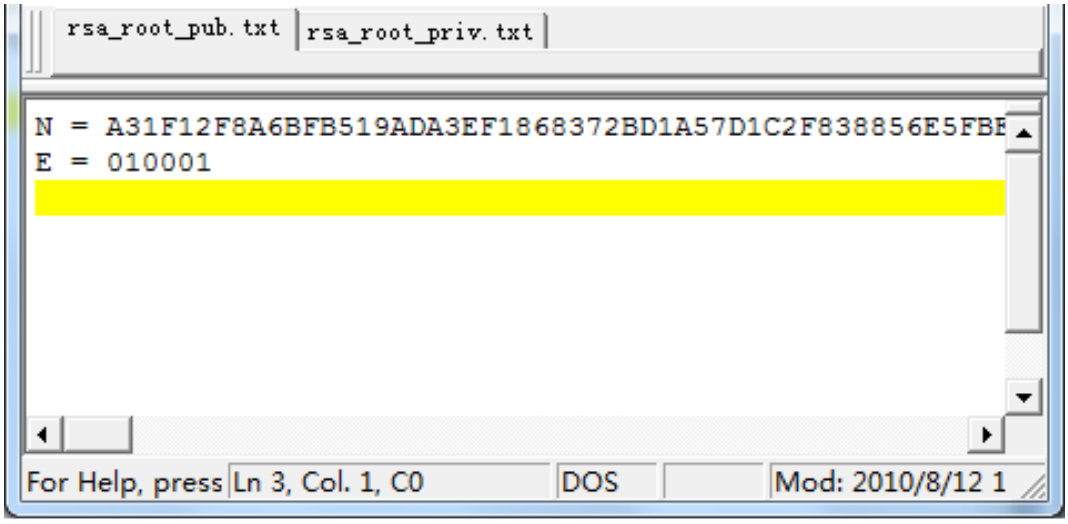
- root 公钥
- root 私钥
- external 公钥

- external 私钥。

root 公钥需要烧写到芯片的 OTP 区域，请参考开发包中 ca_writeRSAkey.c 这个示例。

root 私钥，external 公钥和 external 私钥需要给 CASignTool 使用。CASignTool 工具需要公钥的格式如图 2-19 所示。

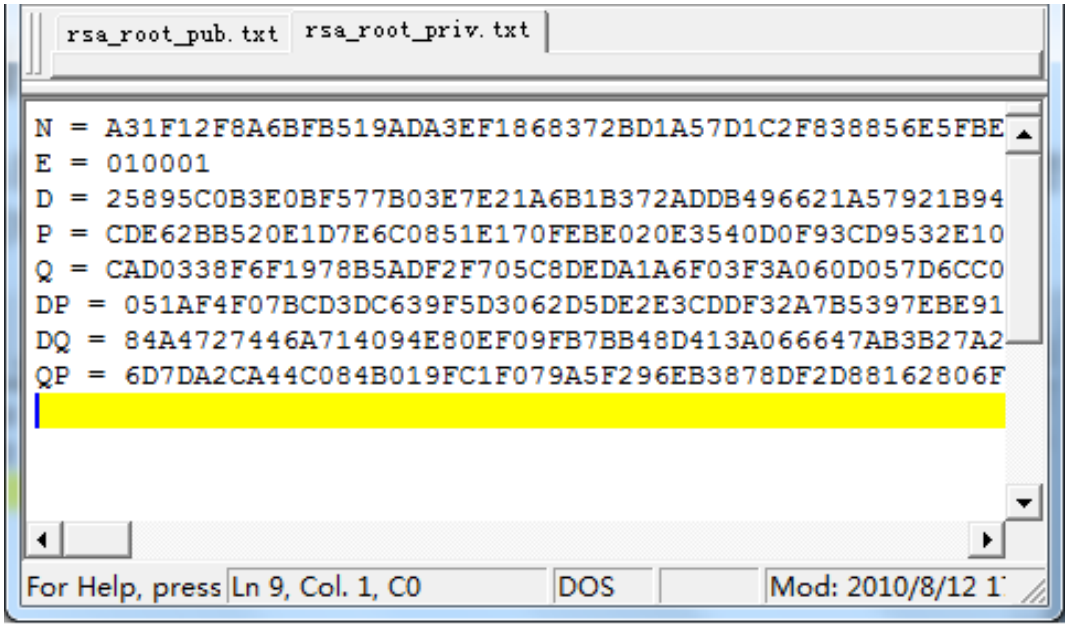
图2-19 CASignTool 工具需要公钥的格式



公钥文件是一个文本文件。N 是一个 2048bit 的 16 进制数，也就是说“N＝ ”后面有 512 个 16 进制数。

CASignTool 工具需要私钥的格式如图 2-20 所示。

图2-20 CASignTool 工具需要私钥的格式



私钥文本也是一个文本文件。

步骤 2 单板配置参数，请参考“2.1.1 环境准备”准备 cfg.bin。

步骤 3 准备普通 BOOT。

----结束

注意

如果客户希望使用海思的签名工具：CASignTool.exe，**那么，客户的** RSA Private Key **必须要包含**(N,E,D,P,Q)5 **组数据。**

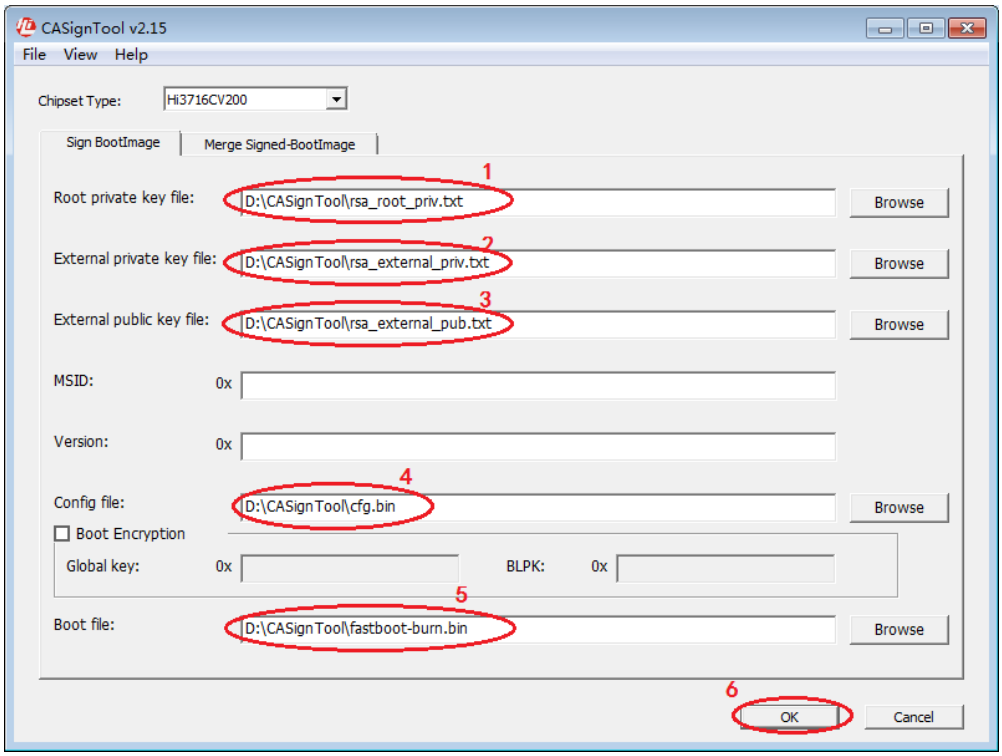
客户也可以参考：PolarSSL (http://www.polarssl.org)，**下载** polarssl **后，修改** polarssl/programs/pkey/rsa_genkey.c，**编译并执行** rsa_genkey。 **可以生成海思所需要的** RSA Key **数据。**

STB **厂商可以使用**“Create RSA Key”**界面产生用于前期开发的** RSA Key。**正式生产时，** CA **公司维护** RSA Private Key，STB **厂商可以向** CA **公司申请** RSA Public Key，**并将待签名** Image **文件提交** CA **公司获得签名。**

2.3.2 操作流程

操作流程如下：

图2-21 自签名 BOOT 流程



步骤 1 选择“2.3.1 环境准备”中准备的 root 私钥。

步骤 2 选择“2.3.1 环境准备”中准备的 external 私钥。

步骤 3 选择“2.3.1 环境准备”中准备的 external 公钥。

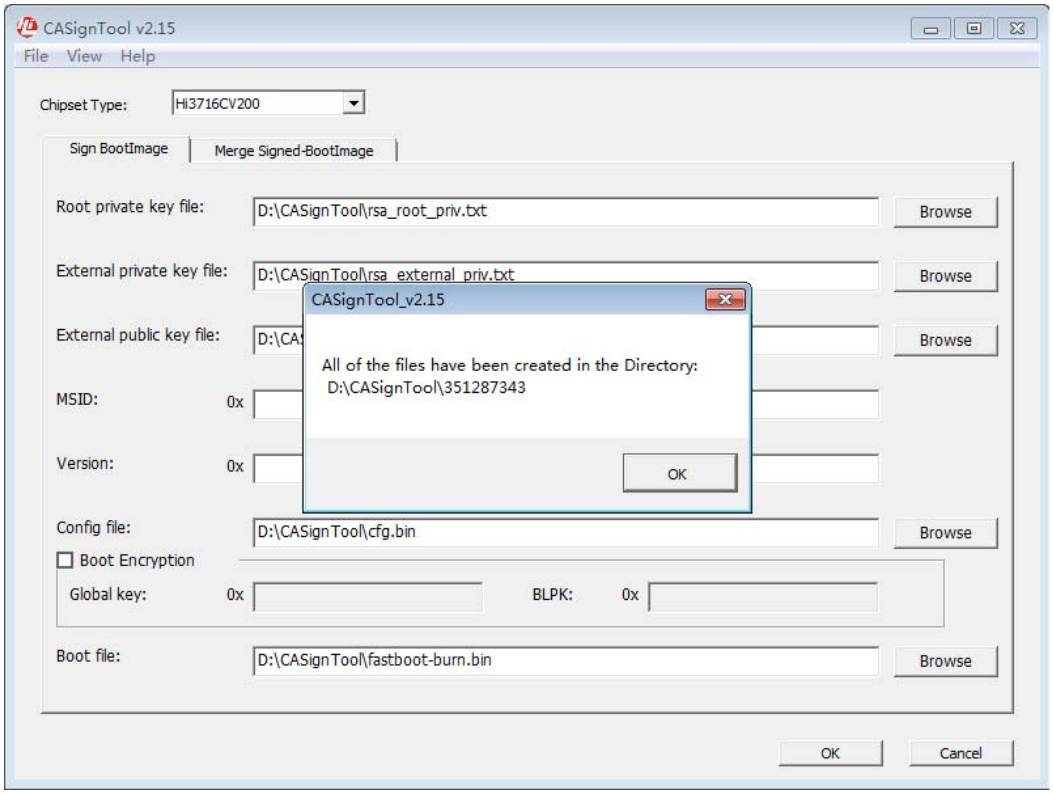
步骤 4 选择“2.3.1 环境准备”中准备的 cfg.bin 文件。

步骤 5 选择“2.3.1 环境准备”中准备的 BOOT 文件。

步骤 6 单击“OK”键。

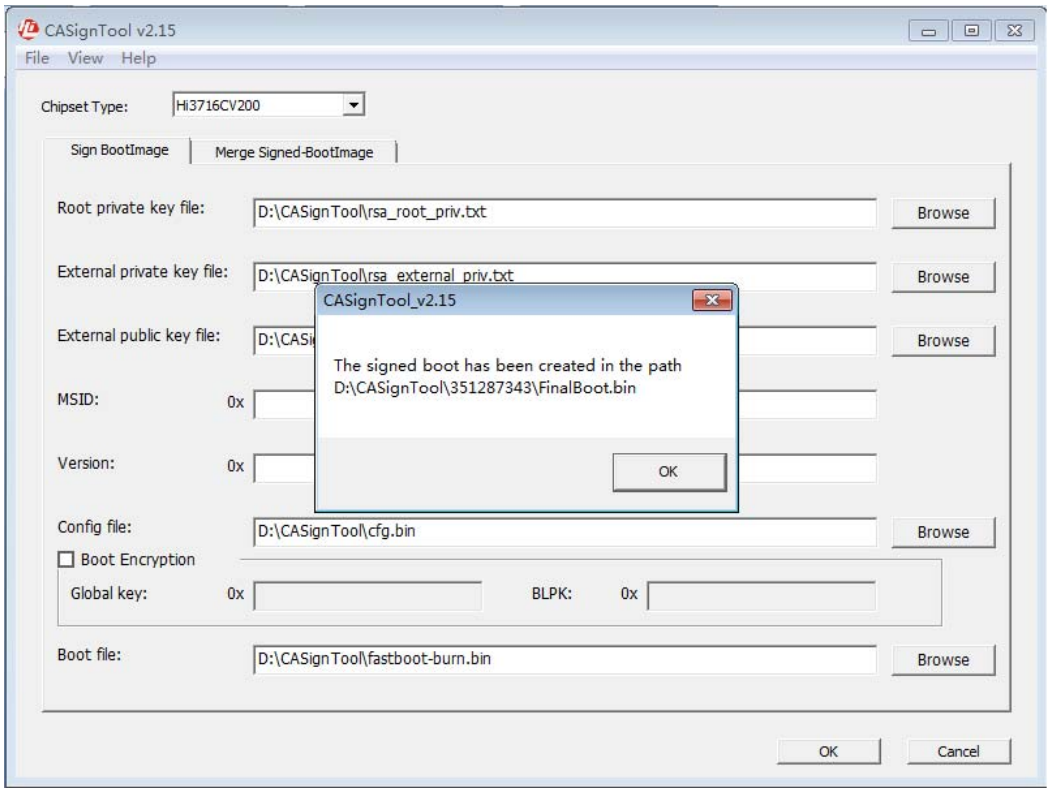
步骤 7 工具将会在工具所在的文件夹新建一个文件夹，生成文件放在这个文件夹下。示例中 351287343 这个文件夹名为随机生成，与实际生成会不一致，请以实际提示为准。

图2-22 生成文件



步骤 8　带签名的安全 BOOT 的名字为 FinalBoot.bin。

图2-23 生成 FinalBoot.bin 文件



----结束

2.3.3 注意事项

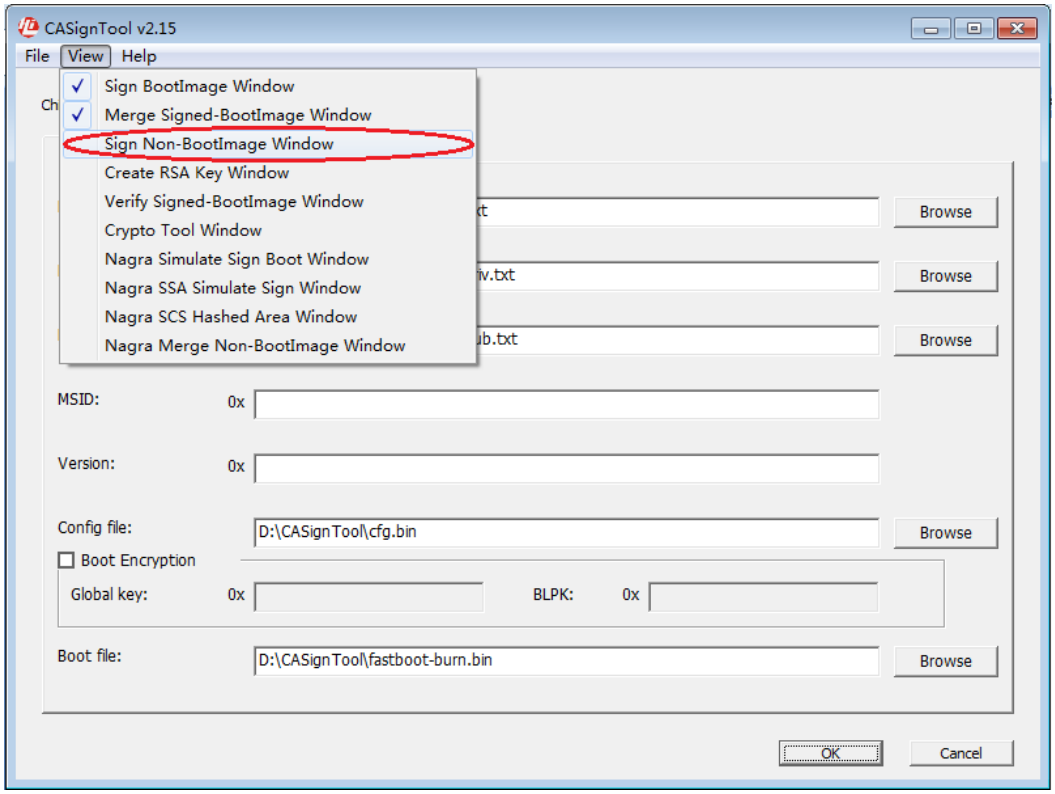
如果需要配置市场区域码、Version ID 和对 BOOT 进行加密，请参考”2.1.3 注意事项”。

2.4 生成非 BOOT 的镜像文件的签名

该功能用于将非 BOOT 的镜像文件，包括 Kernel、文件系统和应用程序等合成签名时要求的格式。并对合成的文件进行签名。

可以按照下图方法打开产生“Sign Non-BootImage”的页面：

图2-24 “Sign Non-BootImage”的页面



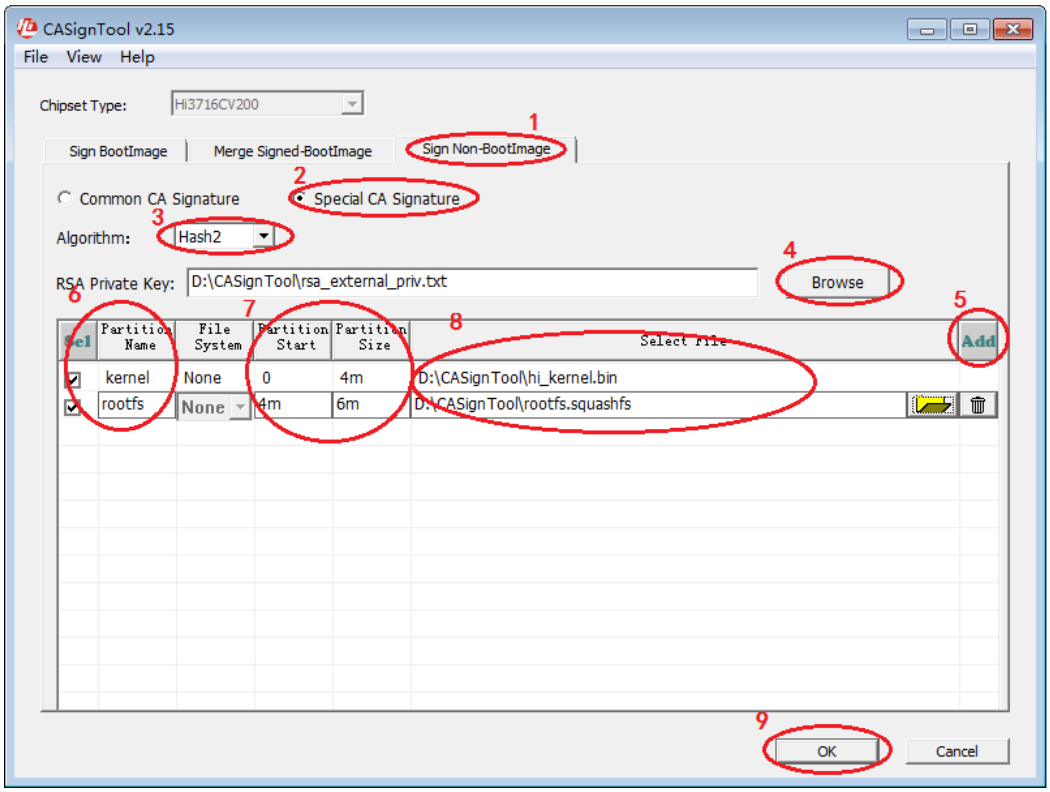
海思提供了 2 类非 BOOT 镜像文件签名方式：非 BOOT 文件通用签名模式和非 BOOT 文件特定签名模式。运用场景请参考《Advanced Secure CA Development Guide.pdf》。

2.4.1 非 BOOT 文件特定签名模式

此类签名主要用在镜像文件必须加密存储的情况。需要使用内存文件系统的市场。产生的签名数据分为 2 部分，索引信息位于镜像文件前面，其余数据位于镜像文件尾部。

签名数据中的索引信息位于镜像文件前面可以快速查找到签名和 Image 参数。

图2-25 生成签名 Image 文件



其签名数据格式如下：

```
typedef struct hi_CAImgHead_S
{
    HI_U8  u8MagicNumber[32];                 //Magic Number: "Hisilicon_ADVCA_ImgHead_MagicNum"
    HI_U8  u8Version[8];                      //version: "V000 0003"
    HI_U32 u32TotalLen;                       //Total length
    HI_U32 u32CodeOffset;                     //Image offset
    HI_U32 u32SignedImageLen;                 //Signed Image file size
    HI_U32 u32SignatureOffset;                //Signed Image offset
    HI_U32 u32SignatureLen;                   //Signature length
    HI_U32 u32BlockNum;                       //Image block number
    HI_U32 u32BlockOffset[IMG_MAX_BLOCK_NUM];    //Each Block offset
    HI_U32 u32BlockLength[IMG_MAX_BLOCK_NUM];    //Each Block length
    HI_U32 Reserverd[32];
    HI_U32 u32CRC32;                          //CRC32 value
} HI_CAImgHead_S;
```

使用方法如图 2-25 所示：

步骤 1 选择”Sign Non-BootImage”页面。

步骤 2 选择 Special CA Signature。

步骤 3 选择签名算法（支持 Hash1 和 Hash2 算法，默认选择 Hash2）。

步骤 4 选择签名使用的 RSA Private key 文件。

步骤 5 添加需要签名的文件列表。

步骤 6　选择需要用于生成签名的文件。

步骤 7　设置文件在 Flash 分区中的（相对）起始地址和长度。

说明

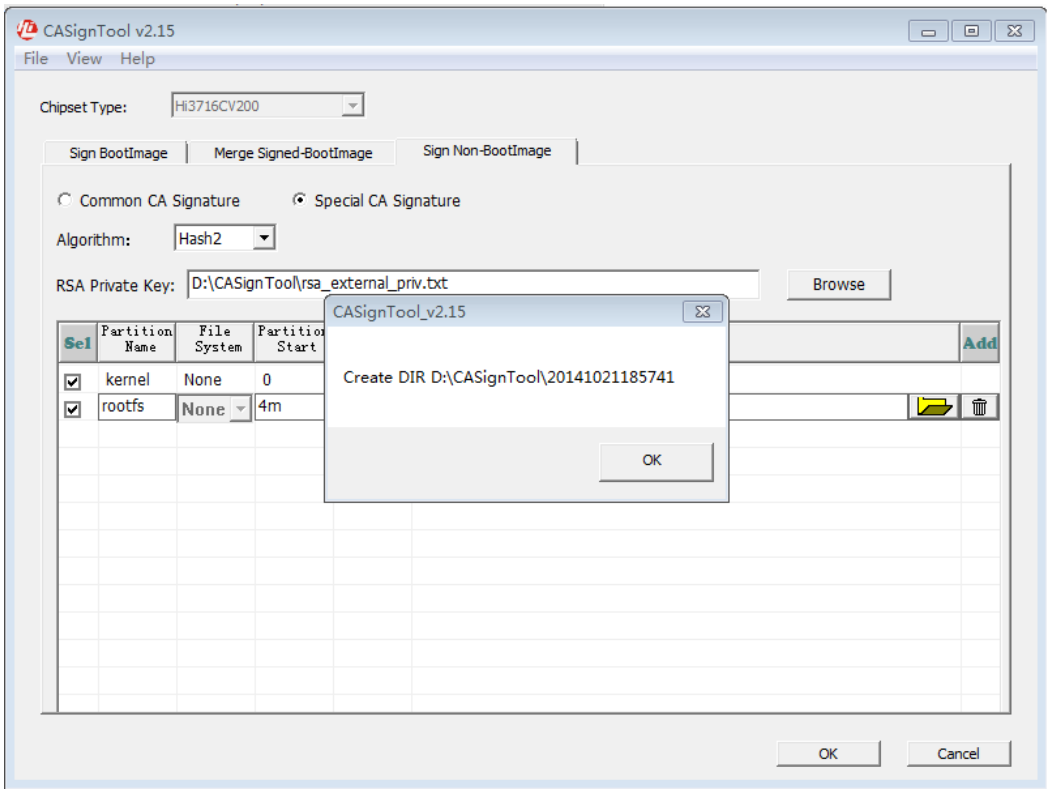
- 第一个文件的相对起始地址必须填 0。
- 单位为 K 或者兆（M）。
- Partition Start 必须大于或等于上一个分区的 Partition Start+PartionSize。
- 签名之后镜像的大小依赖于最后一个分区镜像的实际大小，而与最后一个分区长度无关。

步骤 8　添加需要签名的文件列表。

- 如果只勾选了单一文件，则生成此文件签名后的最终镜像；
- 如果勾选多个文件，则先将这些文件合并，然后对合并后的镜像做签名。

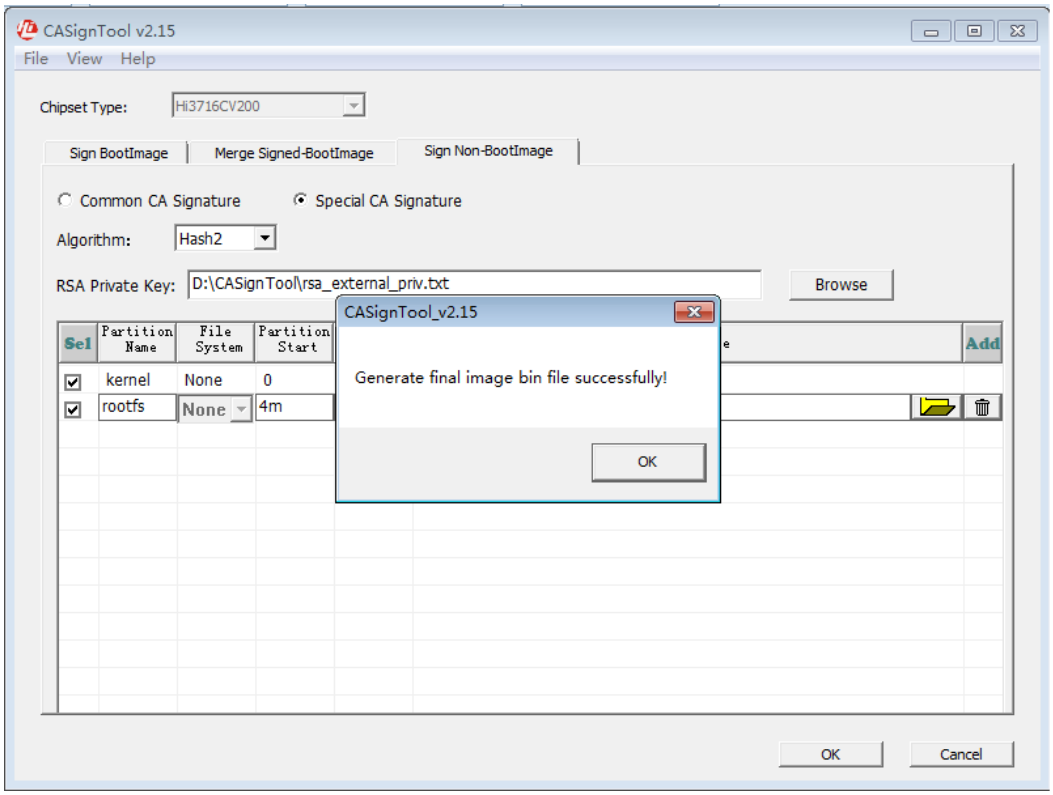
步骤 9　点击 OK，即可在 CASignTool 下产生一个新文件夹。

图2-26 生成最终签名镜像过程提示



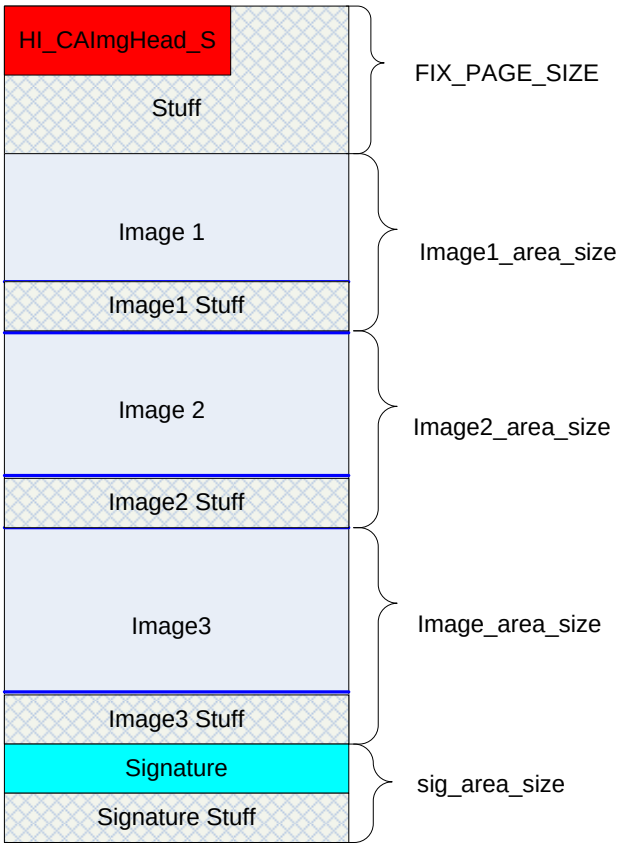
按确定后可以生成最终签名镜像 FinalImage.bin，如图 2-27 所示。

图2-27 生成最终签名镜像 FinalImage.bin



最终产生的被签名后的镜像文件结构图如图 2-28 所示。

图2-28 非 BOOT 文件特定签名模式生成最终签名镜像文件结构图



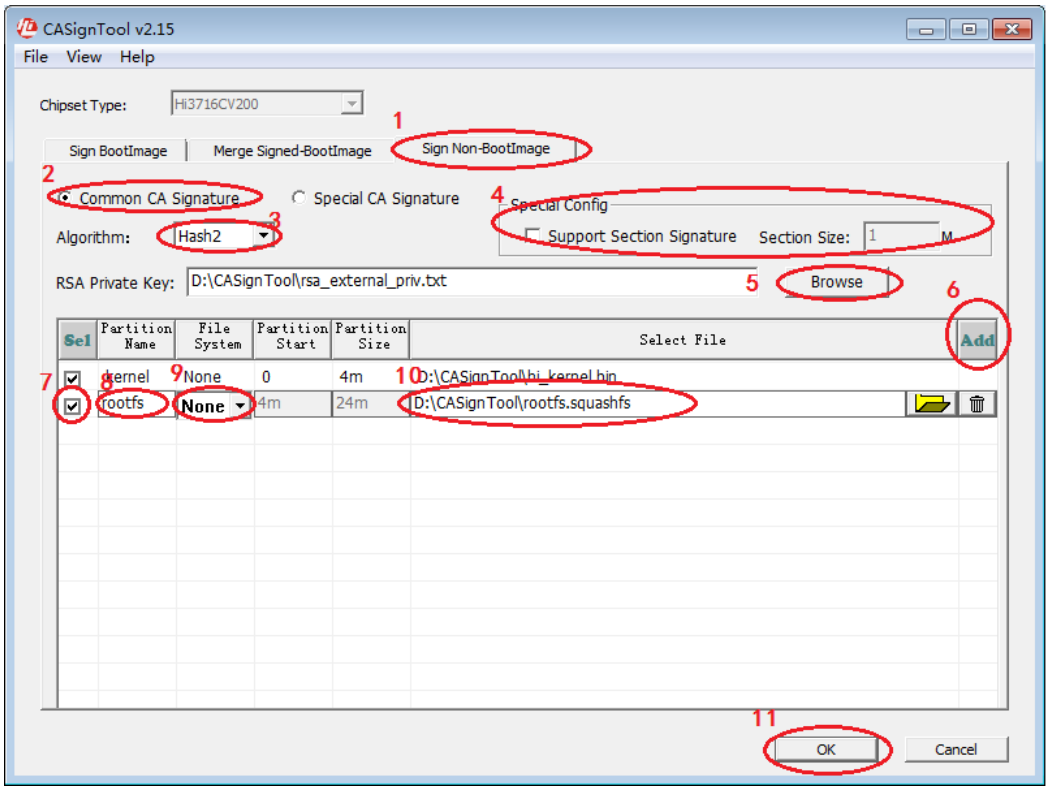
----结束

2.4.2 非 BOOT 文件通用签名模式

此类签名主要用在 Image 文件不需要加密存储的情况。可以使用在非内存文件系统的场景。产生的签名数据位于镜像文件后面。

由于签名数据位于镜像文件的后面，实际使用时，机顶盒应使用其他方式指名其签名的具体位置。

图2-29 生成签名 Image 文件



其签名数据格式如下：

```
typedef struct
{
    unsigned char u8MagicNumber[32];          //Magic Number: "Hisilicon_ADVCA_ImgHead_MagicNum"
    unsigned char u8Version[8];               //version: "V000 0003"
    unsigned int  u32CreateDay;               //yyyymmdd
    unsigned int  u32CreateTime;              //hhmmss
    unsigned int  u32HeadLength;              //The following data size
    unsigned int  u32SignedDataLength;        //signed data length
    unsigned int  u32IsYaffsImage;            //Yaffsr image need to specail read-out, No use
    unsigned int  u32IsConfigNandBlock;        //Yaffsr image need to specail read-out, No use
    unsigned int  u32NandBlockType;           //Yaffsr image need to specail read-out, No use
    unsigned int  u32IsNeedEncrypt;           //if "1", code need to be encrypted.
    unsigned int  u32IsEncrypted;             //if "1", code has encrypted.
    unsigned int  u32HashType;    //if "0", u8Sha save sha1 of code, if "1",  u8Sha save sha256 of code
    unsigned char u8Sha[32];                  //SHA value
    unsigned int  u32SignatureLen;            //Actural Signature length
    unsigned char u8Signature[256];           //Max length:0x100
    unsigned char OrignalImagePath[256];       //Max length:
    unsigned char RSAPrivateKeyPath[256];      //Max length:0x100
    unsigned int  u32CurrentsectionID;         //begin with 0
    unsigned int  u32SectionSize;             //section size
    unsigned int  u32TotalsectionID;           //Max section ID > 1
    unsigned int  CRC32;                      //CRC32 value
} HI_CASignImageTail_S;
```

使用方法如下所示：

步骤 1 选择 “Sign Non-BootImage”页面。

步骤 2 选择 Common CA Signature。

步骤 3 选择签名算法（支持 Hash1 和 Hash2 算法，默认选择 Hash2）。

步骤 4 选择是否支持将镜像文件分为多个块，每个块单独签名。

- 如果勾选支持”Support Section Signature”。需要设置分块大小。默认为 1MB。
- 分块大小最大不能超过镜像文件的 1/2；分块大小最大值为 100MB。
- 这个功能是考虑到被签名的镜像文件非常大(例如，Andriod 的文件系统的大小超过 200MB)，导致校验整个镜像数据签名的时间过长。这种情况下，部分运营商或 CA 公司认为可以采用随机校验若干个数据块的方法。

步骤 5 选择签名使用的 RSA Private key 文件。

步骤 6 添加需要签名的文件列表。

- 如果只勾选了单一文件，则生成此文件签名后的最终镜像；
- 如果勾选多个文件，则每个文件都可能产生：
  - 一个包含 Image 与签名的文件；
  - 一个独立的签名数据文件。

步骤 7 添加需要生成签名的列表项。

步骤 8 设置文件在 Flash 中的分区名。

步骤 9 设置分区类型，(支持 NULL 和 Yaffs2 两种选项，默认为 NULL)。

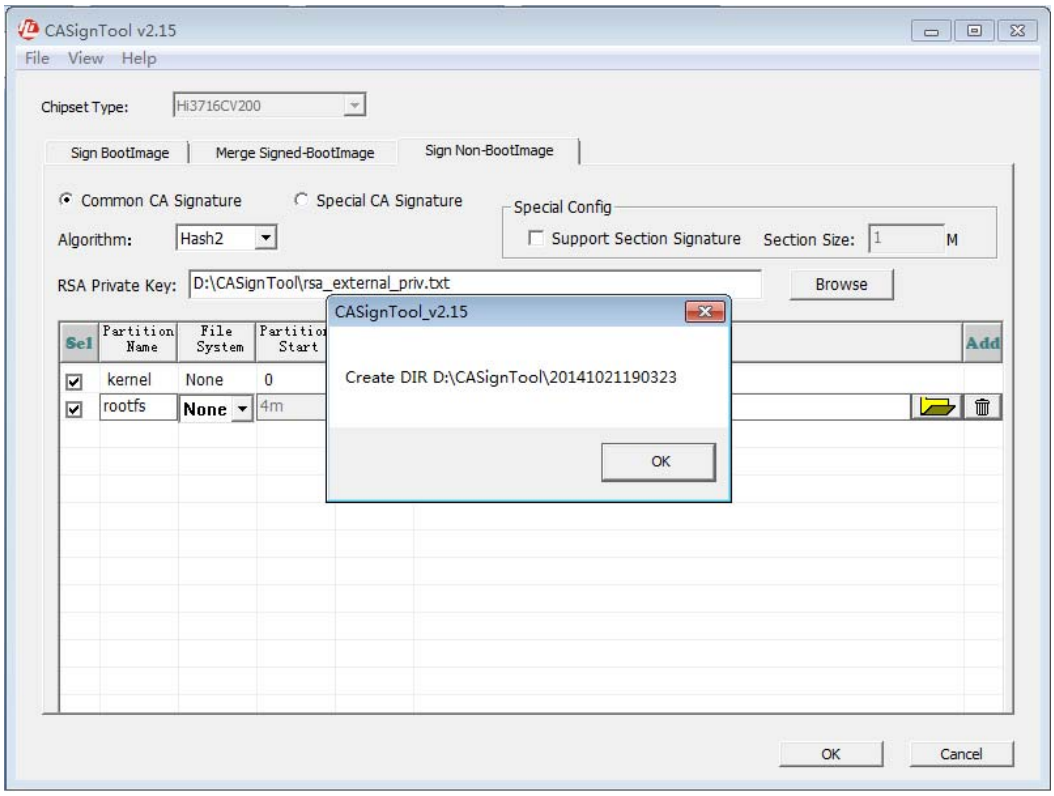
说明

如果选择该分区为 Yaffs2 文件系统分区，被签名的数据将包括 OOB 区域。工具为该 Image 文件只产生独立的签名数据文件。我们不建议客户使用 Yaffs2 文件系统。

步骤 10 添加需要签名的镜像文件。

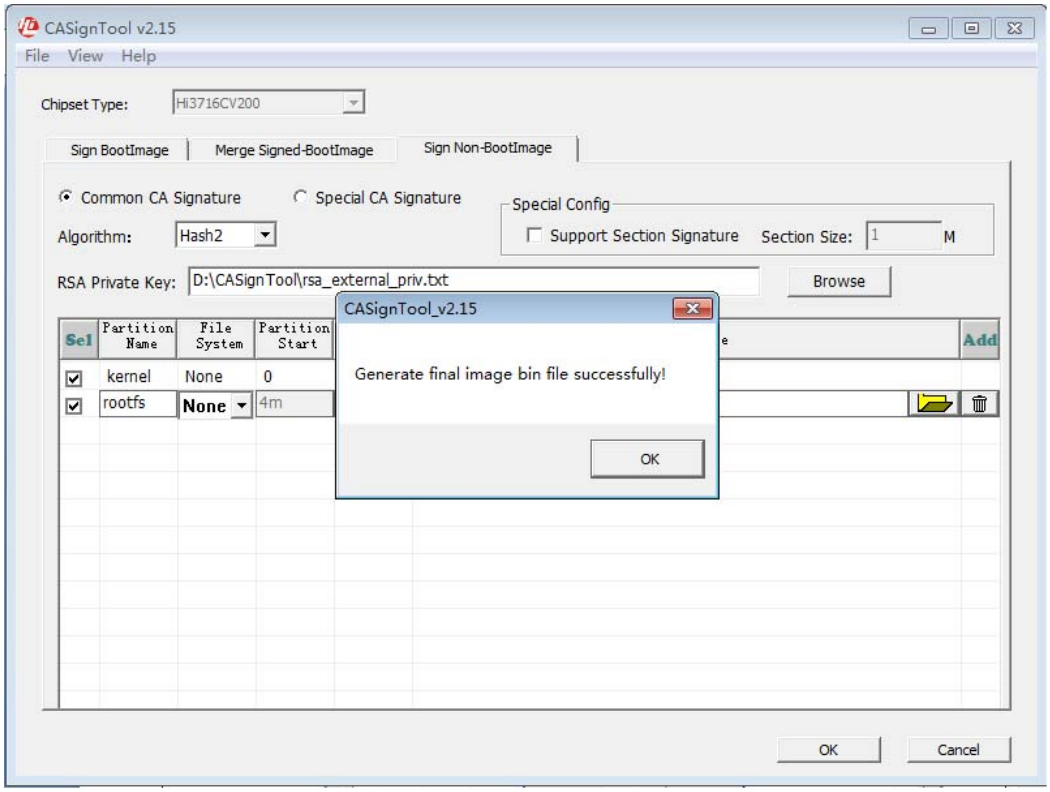
步骤 11 点击 OK，即可在 CASignTool 下产生一个新文件夹。

图2-30 生成签名镜像过程提示



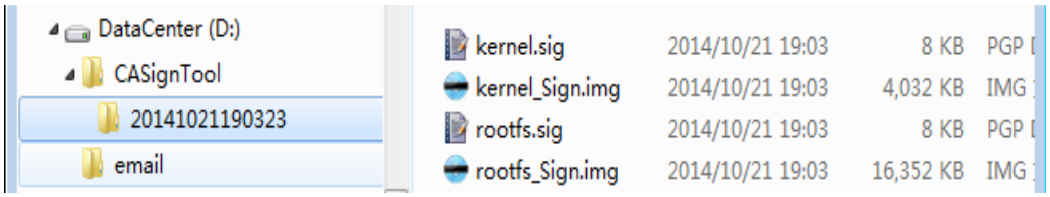
按确定后生成最终签名镜像 FinalImage.bin，如图 2-31 所示。

图2-31 生成最终签名镜像.



产生的签名文件列表如图 2-32 所示。

图2-32 签名文件列表



最终产生的被签名后的镜像文件结构图如图 2-33 和图 2-34 所示。

图2-33 非 BOOT 文件通用签名模式生成最终签名镜像文件结构图(未分块签名)

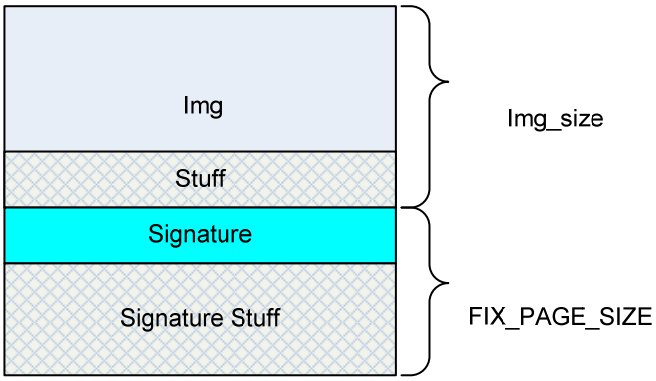
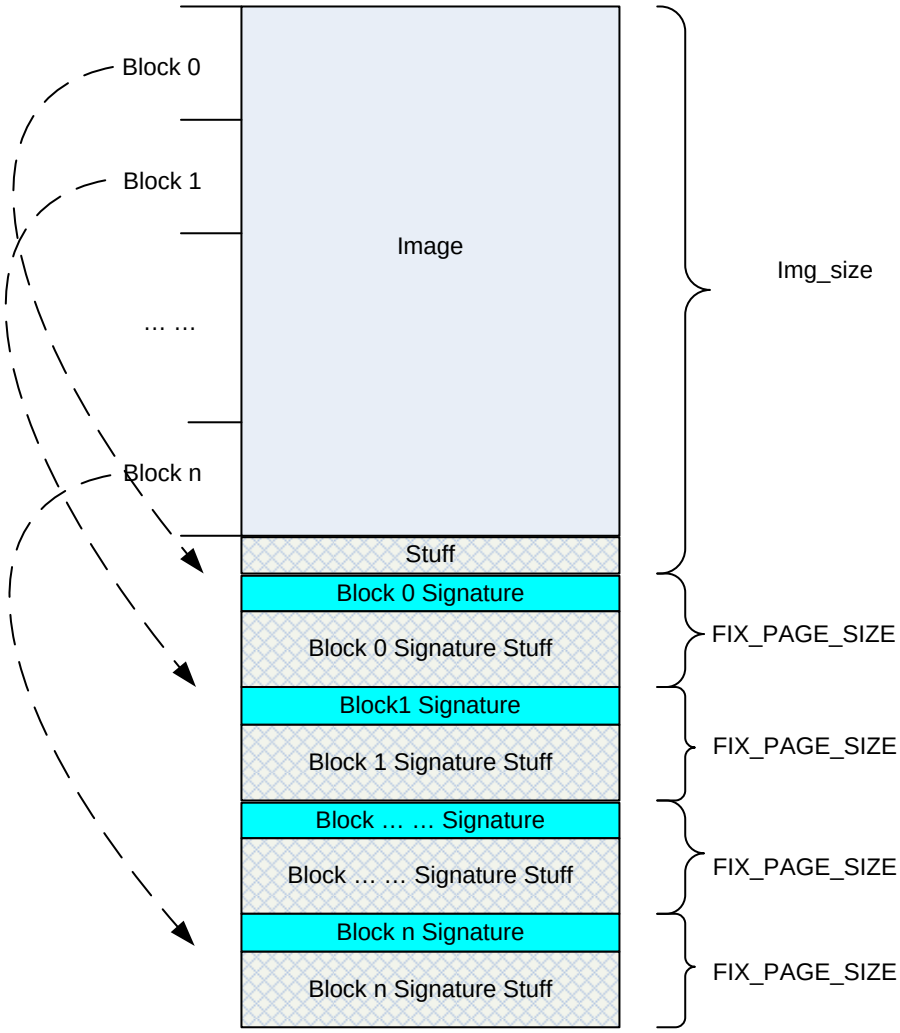


图2-34 非 BOOT 文件通用签名模式生成最终签名镜像文件结构图(分块签名)

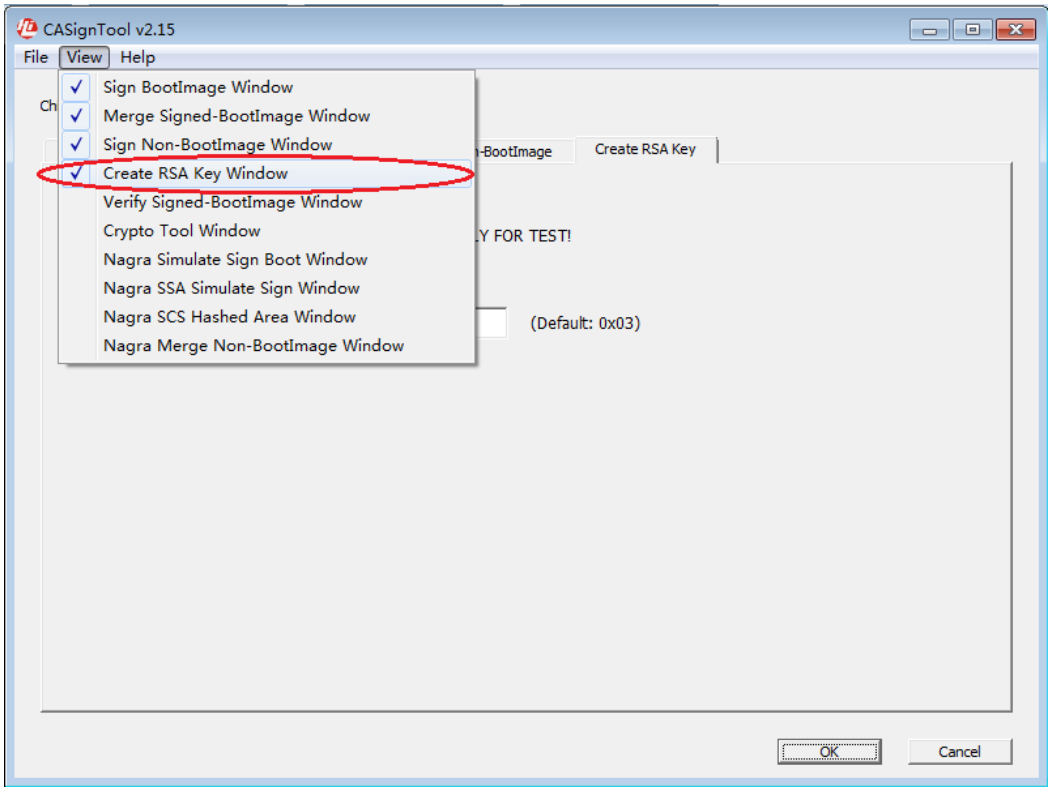


----结束

2.5 生成 RSA 非对称密钥

客户可以通过“Create RSA Key”的页面，产生测试所使用的 RSA 非对称密钥。可以按照下图方法打开产生“Create RSA Key”的页面：

图2-35 “Create RSA Key”的页面



使用方法如下所示：

步骤 1 选择“Create RSA Key”页面。

步骤 2 设置 RSA Key 的 E 值。该值一般为 0x03 或者 0x10001。建议选择 0x03。

步骤 3 点击 OK，即可以产生 RSA 密钥。

图2-36 生成 RSA 非对称密钥

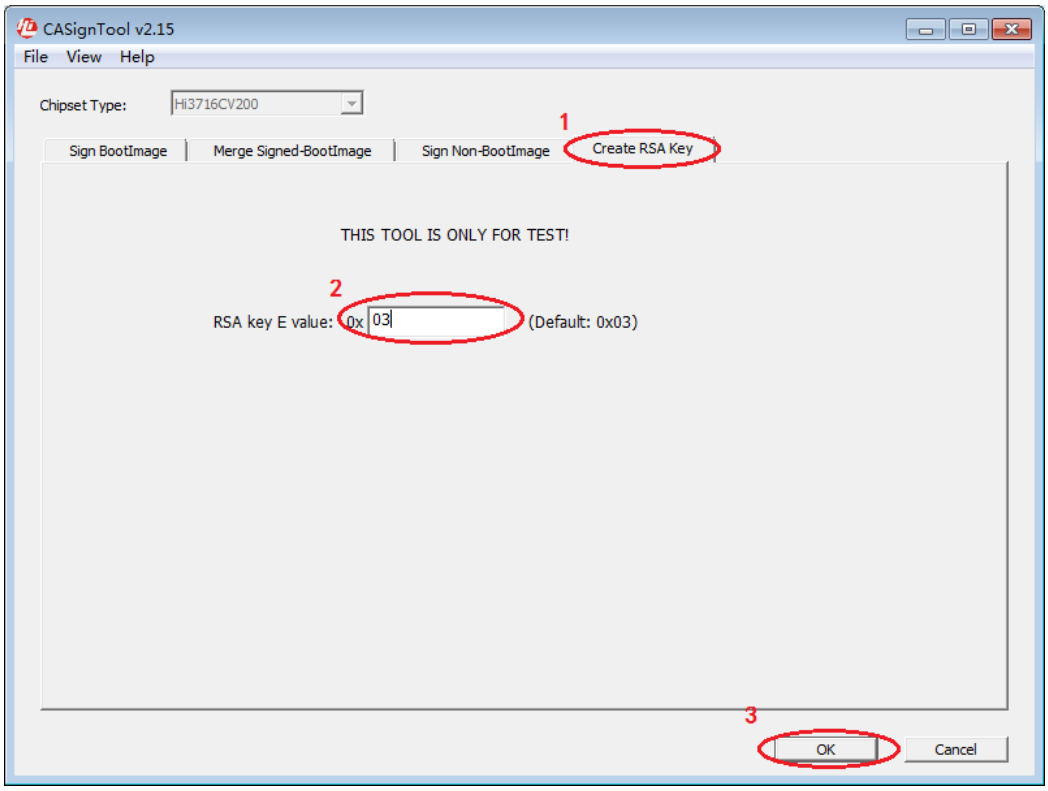
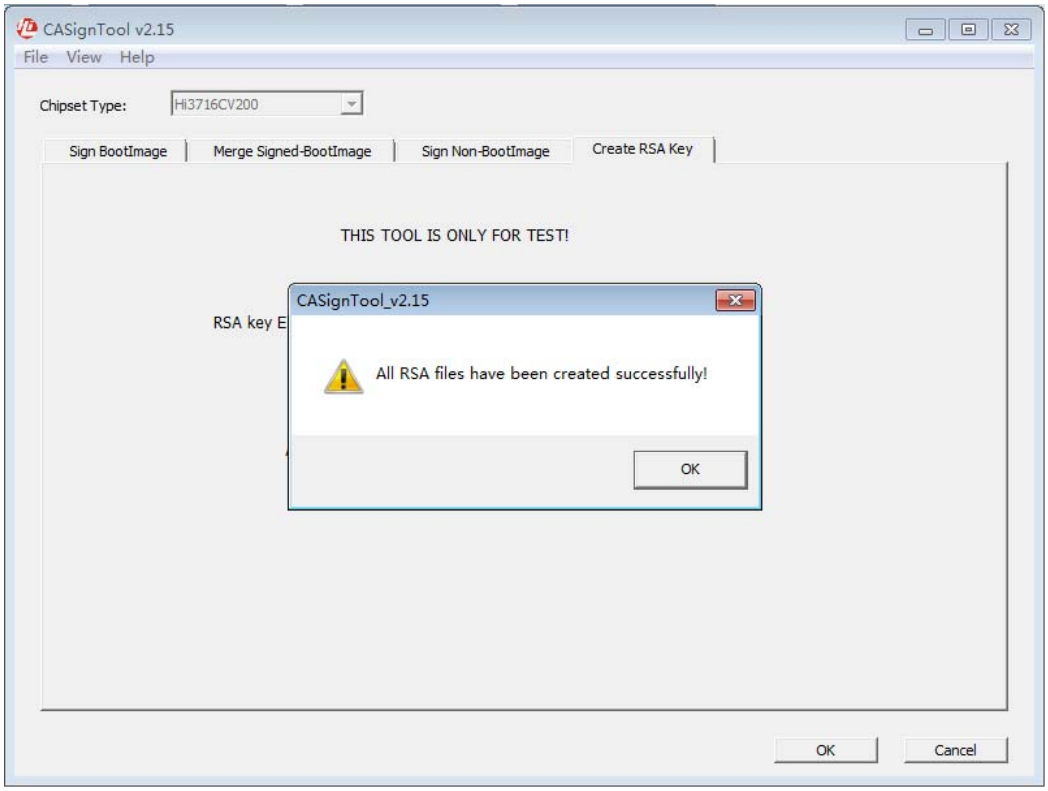
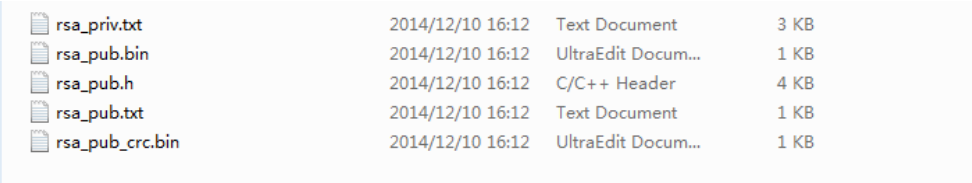


图2-37 成功生成 RSA 非对称密钥提示图



这个工具将会产生文件如图 2-38 图。

图2-38 所生成 RSA 非对称密钥文件



其中，rsa_priv.txt 为 RSA 私钥，文本格式；rsa_pub.txt 为 RSA 公钥，文本格式；rsa_pub.bin 为 RSA 公钥，二进制格式；rsa_pub.h 为 RSA 公钥，C 语言数组格式；rsa_pub_crc.bin 为带 CRC32 校验码的二进制格式的 RSA 公钥。

注意

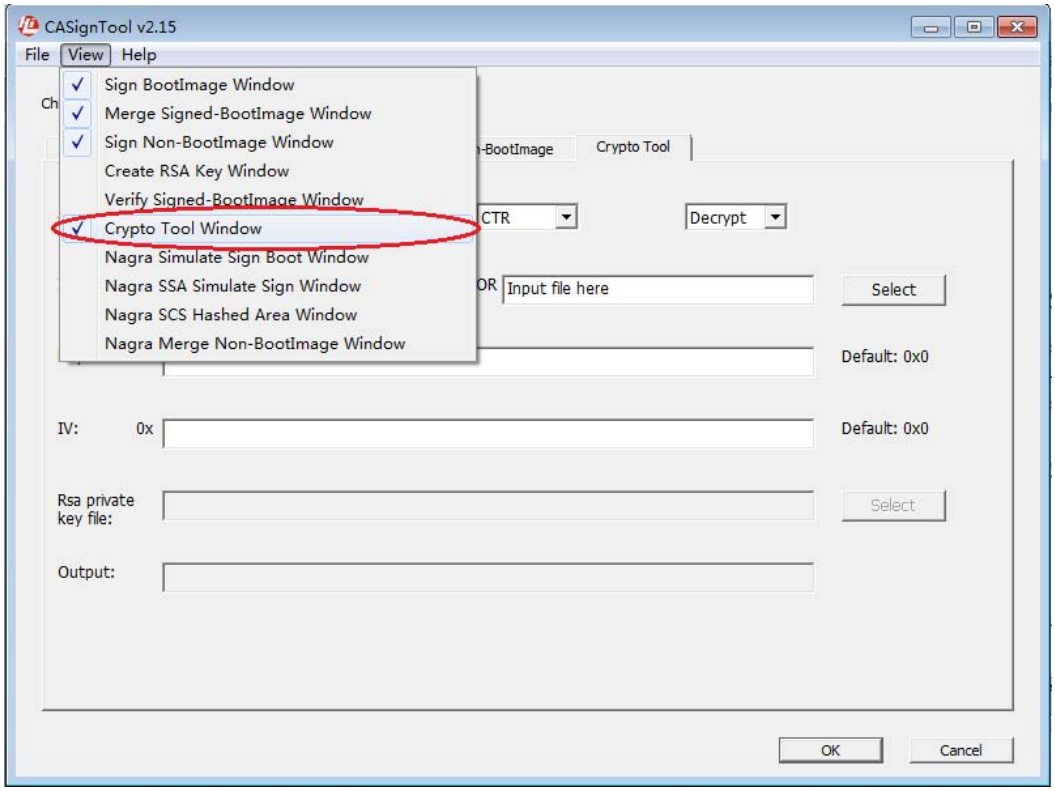
使用 CASignTool 产生的 RSA 密钥可能会出现偶尔产生相同 RSA 密钥的情况。不同的客户有可能使用该工具产生相同的 RSA 密钥。

我们建议机顶盒厂商 CASignTool 产生的 RSA 密钥应用在机顶盒开发测试。

2.6 加解密镜像文件工具

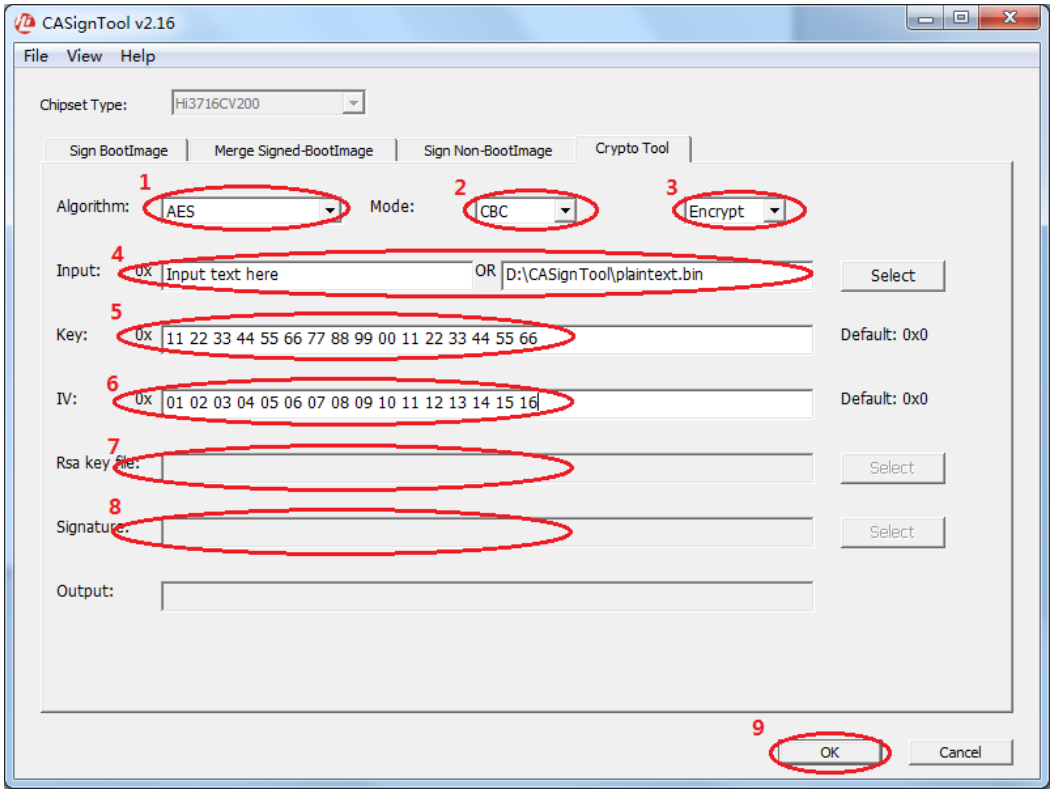
客户可以通过“Crypto Tool”的页面，用于对镜像文件进行加解密操作。可以按照下图方法打开产生“Crypto Tool”的页面：

图2-39 打开“Crypto Tool”页面



客户可以参考如下操作使用该工具。

图2-40 设置“Crypto Tool”页面



使用“Cryto Tool”方法如下所示：

步骤 1 选择算法。可以选择的算法为：AES, TDES, SHA1, SHA256, HiCRC16，RSA_SIGN，RSA_VERIFY.

步骤 2 选择模式。可以选择的模式为：ECB, CBC，CTR。只有在算法中选择了 AES 或者 TDES 时，该选项才有效。CTR 模式仅对 AES 算法有效。

步骤 3 选择加密或者解密。只有在算法中选择了 AES 或者 TDES 时，该选项才有效。

步骤 4 输入需要进行计算的原数据。可以根据实际需要，选择是二进制文件或者字符串数据。

步骤 5 输入密钥。输入值为是 16 个字节十六进制数据，各个字节之间可以由空格隔开，也可以连续填写。只有在算法中选择了 AES 或者 TDES 时，该选项才有效。

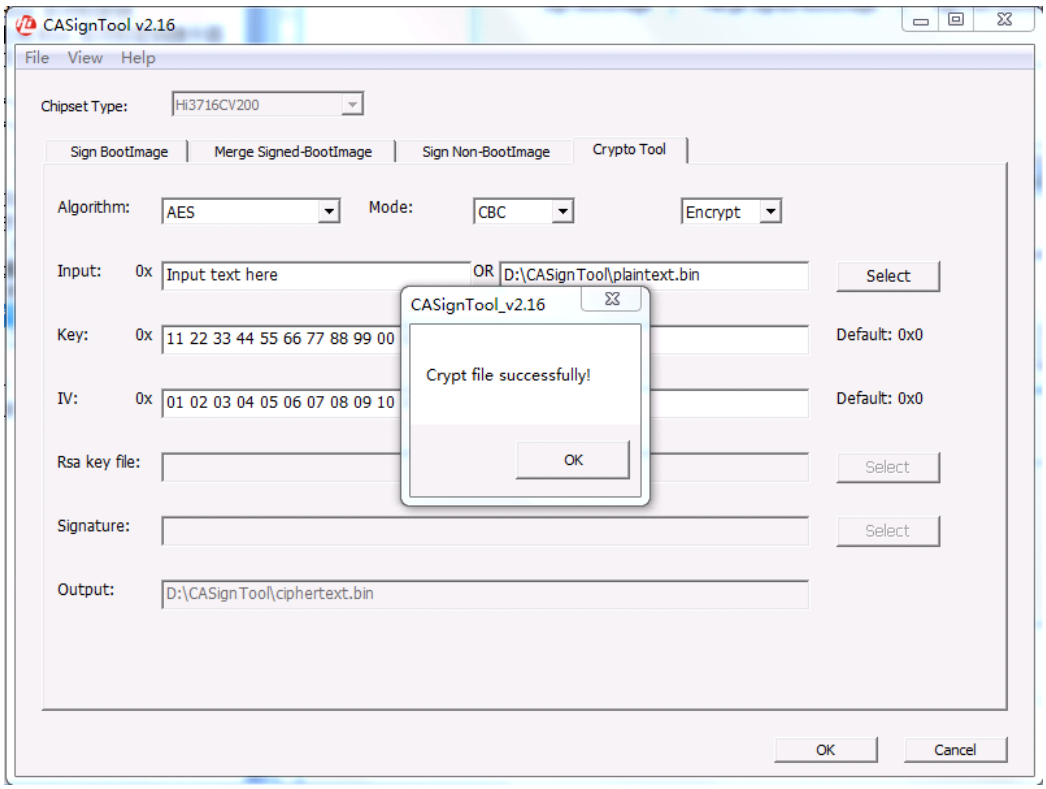
步骤 6 输入 IV 向量。输入值为是 16 个字节十六进制数据，各个字节之间可以由空格隔开，也可以连续填写。只有在算法中选择了 AES 或者 TDES 且模式选择了 CBC 或 CTR 时，该选项才有效。

步骤 7 选择 RSA key 文件。只有在算法中选择了 RSA_SIGN 或者 RSA_VERIFY 时，该选项才有效。如果算法为 RSA_SIGN，请选择 RSA private key 文件；如果算法为 RSA_VERIFY，请选择 RSA public key 文件。

步骤 8 选择签名文件。只有在算法中选择了 RSA_VERIFY 时，该选项才有效。

步骤 9 点击 OK，即可以产生所需要的结果文件。对于 HiCRC16 算法，Output 输出框会显示 CRC16 的计算结果；对于其他算法，Output 输出框会显示所选择的输出文件的完整名称。

图2-41 使用“Crypto Tool”页面操作成功图



----结束

2.7 Verify Signed-BootImage 工具

“Verify Signed-BootImage”工具为海思内部调试工具，机顶盒厂商暂不需要使用该工具。

2.8 Nagra 模拟签名工具

Nagra 模拟签名工具是专门为 Nagra CA 使用。包含：”Nagra Simulate Sign Boot Window”, “Nagra SSA Simulate Sign Window”, “Nagra SCS Hashed Area Window” 和 “Nagra Merge Non-BootImage Window”。

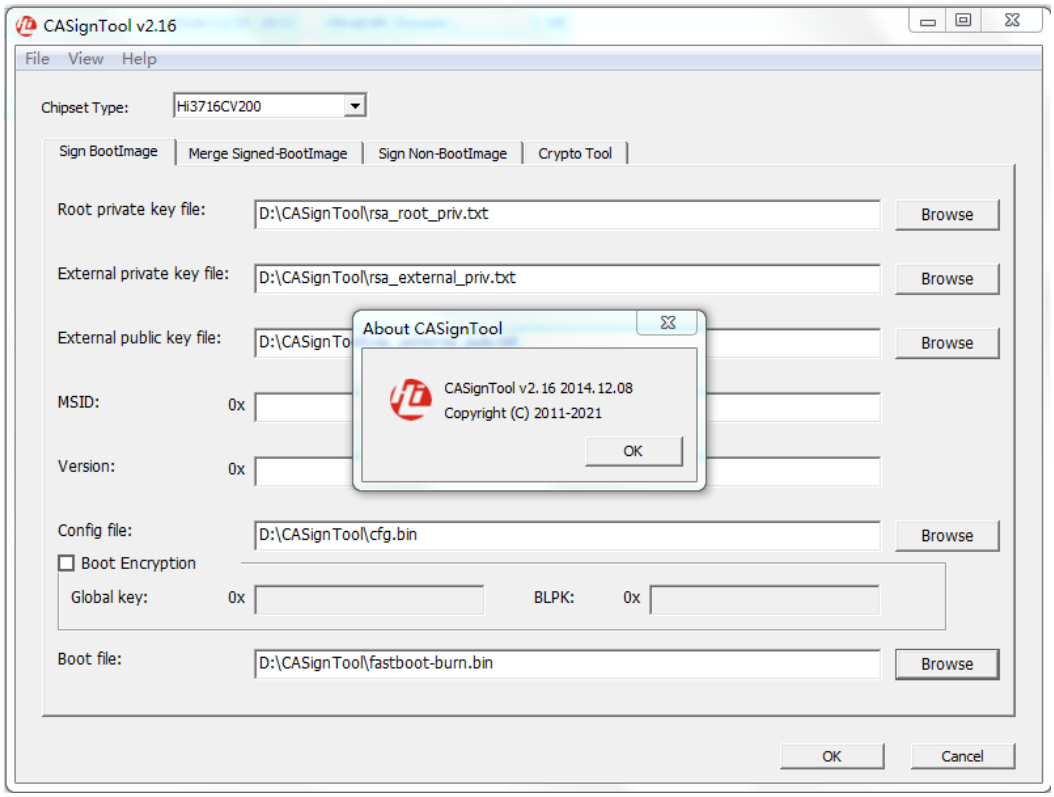
请参考《Nagra CASignTool 使用说明.pdf》获取详细的介绍。

2.9 获取 CASignTool 版本号

请按照如下操作获得 SignTool 的版本号：

请点击菜单上：Help-->About.

图2-42 获取版本号



当前版本号为 2.18。