CV181x/CV180x Secure Boot User Guide

Version： 1.2.5

Release date: 2023-02-06

CONTENTS

Revision History

Revision	Date	Description
0.1	2022/06/01	First Draft
0.2	2022/09/28	Change chip name
0.3	2023/02/01	Update the secure boot usage process
0.4	2023/02/06	CV181x/CV180x document fusion

CHAPTER

ONE

DISCLAIMER

Terms and Conditions

The document and all information contained herein remain the CVITEK Co., Ltd' s ("CVITEK") confidential information, and should not disclose to any third party or use it in any way without CVITEK' s prior written consent.

User shall be liable for any damage and loss caused by unauthority use and disclosure.

CVITEK reserves the right to make changes to information contained in this document at any time and without notice.

All information contained herein is provided in "AS IS" basis, without warranties of any kind, expressed or implied, including without limitation mercantability, non-infringement and fitness for a particular purpose.

In no event shall CVITEK be liable for any third party' s software provided herein, User shall only seek remedy against such third party.

CVITEK especially claims that CVITEK shall have no liable for CVITEK' s work result based on Customer' s specification or published shandard.

Contact Us

Address Building 1, Yard 9, FengHao East Road, Haidian District, Beijing, 100094, China

Building T10, UpperCoast Park, Huizhanwan, Zhancheng Community, Fuhai Street, Baoan District, Shenzhen, 518100, China

Phone +86-10-57590723 +86-10-57590724

Website https://www.sophgo.com/

Forum https://developer.sophgo.com/forum/index.html

CHAPTER

TWO

SECURE BOOT INTRODUCTION

2.1 Image Structure

Figure 2.1 shows the image structure of CV181x/CV180x. When using secure boot, FIP.bin image will be signed and encrypted (encryption function is optional), and the chip will check when boot.

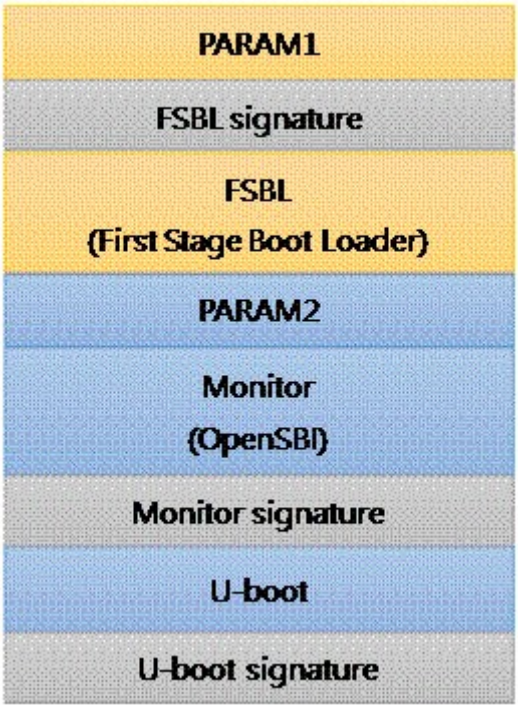
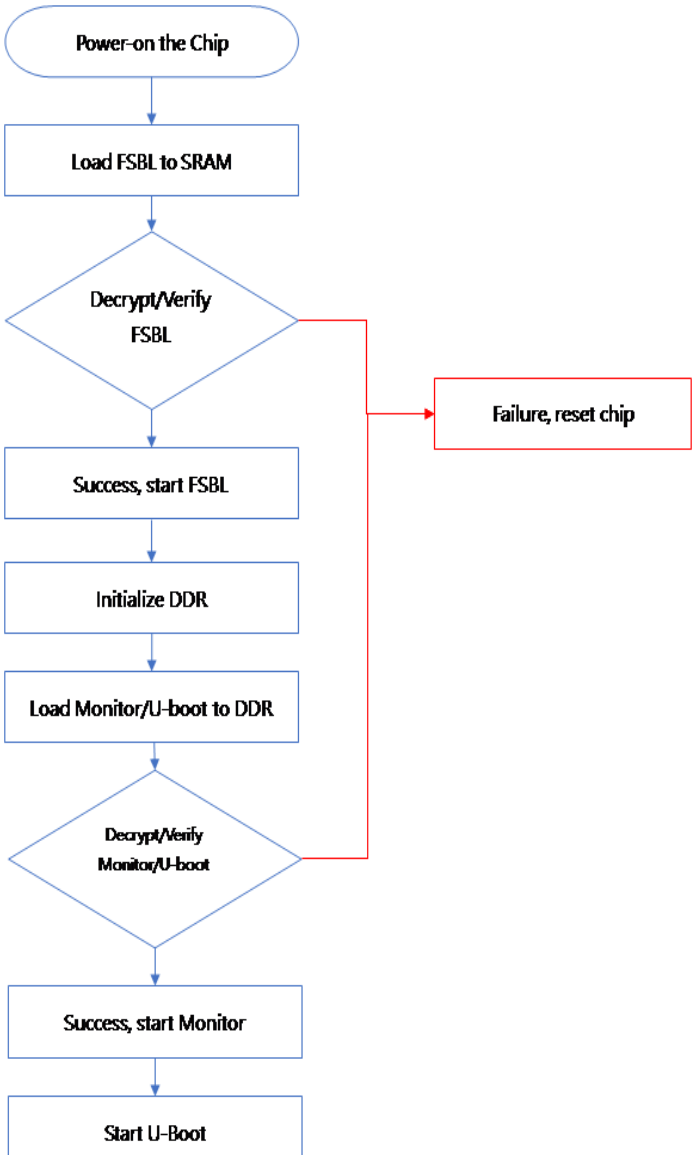


Fig. 2.1: Layout of FIP.bin

2.2 Secure Startup Process

Note: CV180x only supports signature/verification function, please do not use encryption/decryption function, it will cause IC not to start.


Power-on the Chip
Load FSBL to SRAM
Decrypt/Verify FSBL
Failure, reset chip
Success, start FSBL
Initialize DDR
Load Monitor/U-boot to DDR
Decrypt/Verify Monitor/U-boot
Success, start Monitor
Start U-Boot

CHAPTER

THREE

SECURE IMAGE GENERATION

3.1 List of Keys

1.	rsa_hash0.pem	RSA private key for sign FSBL
2.	loader_ek.key	AES key for encrypt FSBL
3.	bl_priv.pem	RSA private key for sign Monitor/u-boot
4.	bl_ek.key	AES key for encrypt Monitor/u-boot

3.2 Generate Keys

1. Generate signature private keys rsa_hash0.pem and bl_priv.pem.

   * RSA keys use 2048 bits and the 4th fermat number.

```
host$ openssl genrsa -out rsa_hash0.pem -F4 2048

host$ openssl genrsa -out bl_priv.pem -F4 2048
```

2. Generate encryption/decryption keys loader_ek.key and bl_ek.key.

   * If signature only without encryption you do not need to generate these keys

   * The following uses random numbers to generate the keys

```
host$ head -c 16 /dev/random > loader_ek.key

host$ head -c 16 /dev/random > bl_ek.key
```

3.3 Sign and Encrypt

3.3.1 Generate Image

Please refer to <U-boot Porting Development Guide> to generate FIP.bin image.

3.3.2 Sign FIP.bin

Note: Precautions

In order to avoid the mass production key being stolen, it is suggested that the mass production key should be kept separately, and the signature tool should be used to sign and encrypt in a secure environment.

Execute the following command to sign the FIP image, fip.bin is the original image, fip_sign.bin is the signed image.

```
cv_crypt$ ./fipsign.py sign \

--root-priv= rsa_hash0.pem \

--bl-priv=bl_priv.pem \

fip.bin fip_sign.bin
```

Tool parameters:

```
cv_crypt$ ./fipsign.py sign

usage:  fipsign.py sign [-h] [--root-priv ROOT_PRIV] [--bl-priv BL_PRIV] SRC_FIP DEST_
↪FIP
```

3.3.3 Sign and encrypt FIP.bin

Execute the following command to sign and encrypt the FIP image, fip.bin is the original image, fip_enc.bin is the signed and encrypted image.

```
cv_crypt$ ./fipsign.py sign-enc \

--root-priv= rsa_hash0.pem \

--bl-priv=bl_priv.pem \

--ldr-ek=loader_ek.key \

--bl-ek=bl_ek.key \

fip.bin fip_enc.bin
```

Tool parameters:

```
cv_crypt$ ./fipsign.py sign-enc

usage: fipsign.py sign-enc [-h] [--ldr-ek LDR_EK] [--bl-ek BL_EK] [--root-priv ROOT_
↪PRIV] [--bl-priv BL_PRIV] SRC_FIP DEST_FIP
```

(continues on next page)

(continued from previous page)

Note: Encryption is optional, if encryption is required, the FIP.bin needs to be configured when compiling.

CONFIG_FSBL_SECURE_BOOT_SUPPORT = y, configuration method:

host$ source build/envsetup_soc.sh

host$ defconfig xxxxxx

host$ menuconfig –> FIP setting –> select [ ] Add secure boot support to FSBL

CHAPTER

FOUR

EFUSE BURNING

Please refer to <eFuse User Guide> to burn eFuse

Note: Precautions

eFuse burning is irreversible. Please make sure the image is signed before executing