# Elftosb for K4W1

## Master Boot image generation

**Cmd call example:**

elftosb.exe -V -d -f k4w1 -J "mb\_ram\_crc\_version.json"

**Json file:**

"family": "k4w1",

"inputImageFile": ".\\workspace\\input\_images\\normal\_boot.bin",

* Input binary image starting with cm33 vector table.

"outputImageExecutionTarget": "ram",

* Where will be the image executed, supported "xip" or "ram".

"outputImageExecutionAddress": "0x1000",

* Address on which will be image executed.

"outputImageAuthenticationType": "crc",

* Security level of generated output image. Supported values are "crc" or "signed". In case of "signed", then all fields specified in chapter 3) certificate block must be added to configuration file.

"outputImageSubtype": "default",

* Image subtype, supported “default” or “NBU” value, if parameter not provided, used “default” value.

"trustZonePresetFile": ".\\workspace\\trustzone\\tztest.bin",

* You can specify file with trust zone registers configuration, which will be included in image and processed by ROM. If empty string is provided as value, or parameter is removed, no preset data included in image.

"isDualBootImageVersion": true,

* Enabling/disabling presence of image version in image header. Feature implemented for dual boot support.

"dualBootImageVersion": "0x1",

* If "isDualBootImageVersion" set to true, then the value is stored in image header as dual boot version. Maximal value is 0xFFFF.

"firmwareVersion": "0x1",

* Value stored in image manifest, checked against monotonous counter in CMPA/CFPA page during boot. Available only when "outputImageAuthenticationType" set to "signed".

"masterBootOutputFile": ".\\workspace\\output\_images\\mb\_ram\_crc\_version.bin"

* Name of the file, where should be stored generated image.

**TEST options:**

Following options are only used for negative test cases generation, not for regular usage.

"testImageManifestMagic": "",

* Replacing image manifest magic value “imgm” by provided value, only value with 4 characters can be used.

"testImageType": false,

* Enabling/disabling bypassing of own image type value.

"testImageTypeValue": 255

* If "testImageType" is set to true, then this value stored as image type on offset 0x24 in image header.

## SB3.1 generation

**Cmd call example:**

elftosb.exe -V -d -f lpc55s3x -j "sb3\_256\_none.json"

**Json file:**

SB3.1 file is always signed, so the all fields from chapter 3) certificate block must be added to configuration file.

"family": "lpc55s3x",

* Niobe4Analog = lpc55s3x.

"containerKeyBlobEncryptionKey": ".\\workspace\\keys\\userkey.txt",

* Path to PCK/NPK key in plain hex string format.

"isNxpContainer": false,

* Internal usage only, used for generating SB files with NXP content e.g. provisioning firmware, sentinel firmware....

"description": "sb3\_256\_256.sb3

* Description up to 16 characters, longer will be truncated by elftosb, stored in SB3.1 manifest. If not provided, inserted elftosb version automatically.

"kdkAccessRights": 3,

* Accepted values are 0, 1, 2 and 3. Value used as key properties for key derivation process, more details can be in CSSv2 manual. Once will be development finished, one fixed value will be used and probably not required as user input.

"containerConfigurationWord": "0x0",

* Flag value in SB3.1 manifest, not used by niobe4analog ROM. Value can be keep 0, or item removed from json file.

"firmwareVersion": "0x1",

* Value stored in SB3.1 manifest, checked against monotonous counter in CMPA/CFPA page during boot.

"sb3BlockOutput": true

* If true, each plain block chunk data section is exported to separated file named with blockNumber\_description.bin. Designed for help with Trust provisioning oracle flow.

**TEST options:**

"testSb3Magic": "",

* Replacing SB3.1 manifest magic value “sbv3” by provided value, only value with 4 characters can be used.

"testSb3ImageType": false,

* Enabling/disabling bypassing of own SB3.1 image type value.

"testSb3ImageTypeValue": 255,

* If " testSb3ImageType " is set to true, then this value stored as image type on offset 0x24 in SB3.1 manifest.

"isEncrypted": false,

* If false, generated SB3.1 blocks are not encrypted, only for testing, ROM cannot accept unencrypted SB3.1 file. If option is missing, SB3.1 file is automatically encrypted.

**Commands:**

Commands details are more described in document SB3Commands.docx available in the same location.

Presented commands syntax available since elftosb 5.2.7.10.

## "commands": [

## {"erase": {"address": "0x1384", "size": "0xFFFF"}},

## {"erase": {"address": "0x2588", "size": "0xFFFF"}},

## {"load": {"address": "0x1256", "file": ".\\workspace\\input\_images\\test1.bin"}},

## {"load": {"address": "0x2588", "file": ".\\workspace\\input\_images\\test2.bin", "authentication": "none"}},

## {"load": {"address": "0x1256", "file": ".\\workspace\\input\_images\\test1.bin", "authentication": "hashlocking"}},

## {"load": {"address": "0x2588", "file": ".\\workspace\\input\_images\\test2.bin", "authentication": "cmac"}},

## {"loadKeyBlob": {"offset": "0x1256", "file": ".\\workspace\\input\_images\\test1.bin", "wrappingKeyId": "NXP\_CUST\_KEK\_INT\_SK"}},

## {"loadKeyBlob": {"offset": "0x2588", "file": ".\\workspace\\input\_images\\test2.bin", "wrappingKeyId": "NXP\_CUST\_KEK\_EXT\_SK"}},

## {"programFuses": {"address": "0x1384", "values": "0x138498, 0x0, 0x5, 0x1ab, 0x1ab, 0xffffffff, 0xfffffff1"}},

## {"programFuses": {"address": "0x2588", "values": "0x138498, 0x0, 0x5, 0x1ab, 0x1ab, 0xffffffff, 0xfffffff1"}},

## {"programIFR": {"address": "0x1384", "file": ".\\workspace\\input\_ifr\\ifr1.bin"}},

## {"programIFR": {"address": "0x2588", "file": ".\\workspace\\input\_ifr\\ifr2.bin"}},

## {"call": {"address": "0x1384"}},

## {"call": {"address": "0x2588"}},

## {"execute": {"address": "0x1384"}},

## {"configureMemory": {"memoryId": "0xA", "configAddress": "0x1842"}},

## {"load": {"address": "0x25", "values": "0x138498"}},

## {"load": {"address": "0x25", "values": "0x138498, 0x25, 0x4856974"}},

## {"fillMemory": {"address": "0x25", "pattern": "0xFFFFFFFF", "size": "0xFF"}}

## ],

## Certificate block

"rootCertificate0File": ".\\workspace\\keys\_certs\\ec\_secp256r1\_cert0.pem",

* Path to root x509 certificate, number 0 is mandatory.

"rootCertificate1File": ".\\workspace\\keys\_certs\\ec\_secp256r1\_cert1.pem",

* Path to root x509 certificate, number 1 is optional. Gaps are not allowed between certificates.

"rootCertificate2File": ".\\workspace\\keys\_certs\\ec\_secp256r1\_cert2.pem",

* Path to root x509 certificate, number 2 is optional. Gaps are not allowed between certificates.

"rootCertificate3File": ".\\workspace\\keys\_certs\\ec\_secp256r1\_cert3.pem",

* Path to root x509 certificate, number 3 is optional. Gaps are not allowed between certificates.

"rootCertificateEllipticCurve": "secp256r1",

* Elliptic curve type of root certificates, all need to be of the same type. accepted values are "secp256r1" and "secp384r1" for niobe4analog.

"mainRootCertId": 0,

* Number of selected root certificates. Accepted values are 0, 1, 2 and 3. If root certificate 0 and 1 are only provided, then only 0 and 1 can be used.

"mainRootCertPrivateKeyFile": ".\\workspace\\keys\_certs\\ec\_pk\_secp256r1\_cert0.pem",

* Path to private key corresponding with selected root certificate.

"useIsk": true,

* If set to true, ISK certificate section included in certificate block and following inputs are required. If false, ISK section is not included in certificate block and SB3.1 block0 is signed directly by selected root certificate. If parameter is missing, used default value as true.

"signingCertificateFile": ".\\workspace\\keys\_certs\\ec\_secp256r1\_sign\_cert.pem",

* Path to x509 certificate, used for SB3.1 block0 signature verification when ISK is used.

"signingCertificatePrivateKeyFile": ".\\workspace\\keys\_certs\\ec\_pk\_secp256r1\_sign\_cert.pem",

* Path to private key file corresponding with public key in "signingCertificatePrivateKeyFile".

"signingCertificateConstraint": "0x0",

* Not used by ROM on niobe4analog, item can be removed from json or set with zero value.

"iskCertificateEllipticCurve": "secp256r1",

* Elyptic curve type of ISK certificate. Accepted values are "secp256r1" and "secp384r1" for niobe4analog. "iskCertificateEllipticCurve" can be same size or lower than "rootCertificateEllipticCurve".

"signCertData": ".\\workspace\\input\_images\\testfffffff.bin",

* Optional, path to user data, which will be included in ISK certificate, not used by ROM.

**TEST options:**

Following options are only used for negative test cases generation, not for regular usage.

"testCertBlockMagic": "",

* Replacing certificate block magic value “chdr” by provided value, only value with 4 characters can be used.

"testCorruptRkhRecord": false,

* Enabling/disabling corrupting of RKTH record.

"testCorruptRkhRecordId": 0,

* If "testCorruptRkhRecord" is set to true, then record corresponding to provided number in RKTH table is corrupted. Enabled values are 0, 1, 2, 3. Please specify only value of specified root certificates, e.g. if root certificate 0 and 1 are only provided, then only 0 and 1 can be used.

"testCorruptIskSignature": false,

* If set to true, then is ISK certificate signature corrupted.