ASR IoT Series

Security Application Notes

Version 1.0.0

Issue Date 2023-09-08

About This Document

This document introduces the usage of all the peripherals of ASR IoT series chips (ASR5502X, ASR5822X, and ASR5952X).

Intended Readers

This document is mainly for engineers who use this chip to develop their own platform and products, for instance:

- PCB Hardware Development Engineer
- Software Engineer
- Technical Support Engineer

Included Chip Models

This document applies to ASR IoT series chips (ASR5502X, ASR5822X, and ASR5952X).

Copyright Notice

Trademark Statement

Electrostatic Discharge (ESD) Warning

This product can be damaged by Electrostatic Discharge (ESD). When handling with this device, the people should be very careful to conduct the ESD protection to avoid any device damage caused by ESD event.

Disclaimer

ASR do not give any warranty of any kind and may make improvements and/or changes in this document or in the product described in this document at any time.

This document is only used as a guide, and no contents in the document constitute any form of warranty. Information in this document is subject to change without notice.

All liability, including liability for infringement of any proprietary rights caused by using the information in this document is disclaimed.

ASR Microelectronics Co., Ltd.

Address: 9F, Building 10, No. 399 Keyuan Road, Zhangjiang High-tech Park, Pudong New Area, Shanghai, 201203, China

Homepage: http://www.asrmicro.com/

Revision History

Date	Version	Release Notes
2023.09	V1.0.0	First release.

Table of contents

1. AES

1.1 Functional Description

AES (Advanced Encryption Standard) algorithm is one of symmetric encryption algorithms which uses the same keys to handle plaintext and cipher.

1.2 APIs

1.2.1 asr_AesInit

Prototype	int asr_AesInit (AesUserContext_t *pContext, AesEncryptMode_t encryptDecryptFlag, AesOperationMode_t operationMode, AesPaddingType_t paddingType).
Function Description	asr_AesInit API assigns a specific user context which will be assigned with data (like encrypt or decrypt flag, operation mode, padding type, AES keys, etc.) of AES algorithm.
Input Parameters	● encryptDecryptFlag: encrypt or decrypt flag. ➢ Encrypt flag -- AES_ENCRYPT ➢ Decrypt flag -- AES_DECRYPT ● operationMode: encrypt or decrypt operation mode. ➢ ECB mode -- AES_MODE_ECB ➢ CBC mode -- AES_MODE_CBC ➢ CBC_MAC mode -- AES_MODE_CBC_MAC ➢ CTR mode -- AES_MODE_CTR ➢ CMAC mode -- AES_MODE_CMAC ● paddingType: when the data Byte length is not a multiple of 16 Bytes, it will fill data according to the paddingType parameter. ➢ No padding type -- AES_PADDING_NONE
Output Parameters	pContext: user context.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00000 -- Parameter pContext is a NULL pointer. ● 0xF00003 -- Parameter operationMode error. ● 0xF00006 -- Parameter EncryptDecryptFlag error. ● 0xF0000E -- Parameter paddingType is not AES_PADDING_NONE. ● 0xF00012 -- Parameter operationMode is AES_MODE_CBC_MAC or AES_MODE_CMAC, and Parameter EncryptDecryptFlag must be AES_ENCRYPT.
Note	None.

1.2.2 asr_AesSetKey

Prototype	int asr_AesSetKey (AesUserContext_t *pContext, AesKeyType_t keyType, void *pKeyData, size_t keyDataSize).
Function Description	asr_AesSetKey API is used to set the key to user context.
Input Parameters	● keyType: AES key type. ➢ user key -- AES_USER_KEY ● PKeyData: it's a structure (AesUserKeyData_t, seeing in Note item) including AES key data and size parameter. ● keyDataSize: the structure's length, which is sizeof (AesUserKeyData_t).
Output Parameters	pContext: user context.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00000 -- Parameter pContext is a NULL pointer. ● 0xF00003 -- User key length is not 16 Bytes. ● 0xF00004 -- Parameter pKeyData or pKeyData->pKey is a NULL pointer. ● 0xF00005 -- Parameter keyType is not AES_USER_KEY.
Note	Structure AesUserKeyData_t prototype is: Typedef struct AesUserKeyData_t { uint8_t *pKey; size_t keySize; } AesUserKeyData_t.

1.2.3 asr_AesSetIv

Prototype	int asr_AesSetIv (AesUserContext_t *pContext, AesIv_t pIV).
Function Description	Set IV value to user context that is needed in CBC, CTR and CBC_MAC mode.
Input Parameters	pIV: IV value must be 16 Bytes.
Output Parameters	pContext: user context.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00000 -- Parameter pContext is a NULL pointer. ● 0xF00001 -- Parameter pIV is a NULL pointer. ● 0xF00002 -- Operation mode is not in CBC, CTR or CBC_MAC mode which is assigned in invoking asr_AesInit API.
Note	None.

1.2.4 asr_AesGetIv

Prototype	int asr_AesGetIv (AesUserContext_t *pContext, AesIv_t pIV).
Function Description	Get IV value from user context in CBC, CTR, and CBC_MAC modes.
Input Parameters	pContext: user context.
Output Parameters	pIV: 16 Bytes IV value.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00000 -- Parameter pContext is a NULL pointer. ● 0xF00002 -- Operation mode is not in CBC, CTR, CBC_MAC or CMAC mode.
Note	None.

1.2.5 asr_AesBlock

Prototype	int asr_AesBlock (AesUserContext_t *pContext, uint8_t *pDataIn, size_t dataInSize, uint8_t *pDataOut).
Function Description	Encrypt or decrypt the first and middle block data.
Input Parameters	● pContext: user context. ● pDataIn: encrypted or decrypted data. ● dataInSize: size of data.
Output Parameters	● pDataOut: output encrypted or decrypted data in ECB, CBC and CTR modes by invoking asr_AesBlock, and output data in CMAC, and CBC_MAC modes by invoking asr_AesFinish.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00000 -- Parameter pContext is a NULL pointer. ● 0xF00008 -- Parameter pDataIn is a NULL pointer. ● 0xF00009 -- Parameter pDataOut is a NULL pointer in ECB, CBC, and CTR modes. ● 0xF0000A -- Parameter dataInSize is illegal (should be multiple of 16 Bytes and should not bigger than 0x10000) .
Note	None.

1.2.6 asr_AesFinish

Prototype	int asr_AesFinish (AesUserContext_t *pContext, size_t dataSize, uint8_t *pDataIn, size_t dataInBuffSize, uint8_t *pDataOut, size_t *dataOutBuffSize).
Function Description	Encrypt or decrypt the last block data.
Input Parameters	● pContext: user context. ● dataSize: size of data. ● pDataIn: input data. ● dataInSize: size of input data.
Output Parameters	● pDataOut: output encrypted or decrypted data in ECB, CBC and CTR modes by invoking asr_AesBlock, and output data in CMAC and CBC_MAC modes by invoking asr_AesFinish. ● dataOutBuffSize: get the output data size.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00000 -- Parameter pContext is a NULL pointer. ● 0xF00008 -- Parameter pDataIn is a NULL pointer and dataSIze is not 0. ● 0xF00009 -- Parameter pDataOut is a NULL pointer in CMAC and CBC_MAC modes. ● 0xF0000A -- Parameter dataSize is illegal (should be a multiple of 16 Bytes) in ECB, CBC, and CBC_MAC modes and padding mode is None. ● 0xF0000C -- Parameter dataInBuffSize is less than dataSIze. ● 0xF0000D -- Parameter dataOutBufferSize is less than 16 Bytes or dataSize. ● 0xF00011 -- Parameter dataOutBuffSize is a NULL pointer.
Note	None.

1.2.7 asr_AesFree

Prototype	int asr_AesFree (AesUserContext_t *pContext).
Function Description	Clean up the user context with value zero.
Input Parameters	pContext: user context.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00000 -- Parameter pContext is a NULL pointer.
Note	None.

2. CCM

2.1 Functional Description

CCM (Counter with Cipher Block Chaining Message Authentication Code) algorithm is applied to message encryption and authentication through AES-CTR and AES-CMAC respectively.

2.2 APIs

2.2.1 asr_AESCCM

Prototype	int asr_AESCCM (aesEncryptMode_t EncrDecrMode, AESCCM_Key_t CCM_key, AESCCM_KeySize_t KeySizeId, uint8_t *N_ptr, uint8_t SizeOfN, uint8_t *ADataIn_ptr, uint32_t ADataInSize, uint8_t *TextDataIn_ptr, uint32_t TextDataInSize, uint8_t *TextDataOut_ptr, uint8_t SizeOfT, AESCCM_Mac_Res_t Mac_Res).
Function Description	Use random data Nonce, keys, additional data to encrypt or decrypt data to get decrypted or encrypted data.
Input Parameters	● EncrDecrMode: encrypt or decrypt mode. ➢ Encrypt mode -- AES_ENCRYPT ➢ Decrypt mode -- AES_DECRYPT ● CCM_key: CCM key. ● KeySizeId: CCM key size, must be 16 Bytes. ● N_ptr: random value nonce pointer. ● SizeOfN: size of nonce pointer space, valid range is from 7 Bytes to 13 Bytes and edge included. ● ADataIn_ptr: input additional data pointer. ● ADataInSize: size of input additional data. ● TextDataIn_ptr: input plaintext or ciphertext pointer. ● TextDataInSize: size of input plaintext or ciphertext. ● SizeOfT: size of input text data, valid values are 4, 6, 8, 10, 12, 14, and 16 Bytes. Mac_Res: Mac result buffer when in decrypt mode.
Output Parameters	● TextDataOut_ptr: output ciphertext or ciphertext pointer. ● Mac_Res: Mac result buffer when in encrypt mode.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF01501 -- Parameter KeySizeId is not 16 Bytes long. ● 0xF01503 -- Parameter EncrDecrMode is invalid. ● 0xF01505 -- Parameter TextDataIn_ptr is a NULL pointer. ● 0xF01506 -- Parameter TextDataOut _ptr is NULL or DataInSize is 0 Byte. ● 0xF01507 -- Parameter TextDataInSize is 0 Byte or greater than AdataSize. ● 0xF01508 -- Parameter TextDataOut _ptr and TextDataIn_ptr overlap in memory space. ● 0xF0150C -- Parameter SizeOfT ir SizeOfN is invalid value. ● 0xF0150D -- Parameter CCM_key or N_ptr is a NULL pointer.
Note	None.

3. ECC

3.1 Functional Description

ECC (Elliptic Curve Cryptography) algorithm is one of asymmetric encryption algorithms.

3.2 APIs

3.2.1 asr_ecpki_getEcDomain

Prototype	const ECPKI_Domain_t *asr_ecpki_getEcDomain (ECPKI_DomainID_t domainId).
Function Description	Get the Elliptic Curve domain parameters , including finite filed value, base point, cofactor and the order of subgroup, according to domainId.
Input Parameters	DomainId: Elliptic Curve domain identity. ➢ EC secp160k1 -- ECPKI_DomainID_secp160k1 ➢ EC secp160r1 -- ECPKI_DomainID_secp160r1 ➢ EC secp160r2 -- ECPKI_DomainID_secp160r2 ➢ EC secp192k1 -- ECPKI_DomainID_secp192k1 ➢ EC secp192r1 -- ECPKI_DomainID_secp192r1 ➢ EC secp224k1 -- ECPKI_DomainID_secp224k1 ➢ EC secp224r1 -- ECPKI_DomainID_secp224r1 ➢ EC secp256k1 -- ECPKI_DomainID_secp256k1 ➢ EC secp256r1 -- ECPKI_DomainID_secp256r1 ➢ EC secp384r1 -- ECPKI_DomainID_secp384r1 ➢ EC secp521r1 -- ECPKI_DomainID_secp521r1
Output Parameters	None.
Return Value	Return value and its meaning: ● 0xXXXX -- The Elliptic Curve domain parameters address according to the domainId. ● NULL -- Parameter domainId is wrong.
Note	0xXXXX means the Elliptic Curve domain parameters address according to the domainId in Return Value item.

3.2.2 asr_ecpki_genkeypair

Prototype	int asr_ecpki_genkeypair (RND_Context_t *pRndContext, const ECPKI_Domian_t *pDomian, ECPKI_UserPrivKey_t *pUserPrivKey, ECPKI_UserPublKey_t *PUserPublKey, ECPKI_KG_TempData_t *pTempBuff, ECPKI_KG_FipsContext_t *pFipsCtx).
Function Description	Generate a pair of private and public keys.
Input Parameters	● pRndContext: a random context buffer. ● pDomain: Elliptic Curve parameters returned by asr_ecpki_getEcDomain. ● pTempBuff: temporary buffer. ● pFipsCtx: FIPS context buffer.
Output Parameters	● pUserPrivKey: private key buffer. ● PUserPublKey: public key buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00802 -- Parameter pDomain is a NULL pointer. ● 0xF00803 -- Parameter pUserPrivKey is a NULL pointer ● 0xF00804 -- Parameter pUserPublKey is a NULL pointer. ● 0xF00805 -- Parameter pTempBuff is a NULL pointer. ● 0xF00806 -- Parameter pRndContext is a NULL pointer. ● 0xF00C01 -- Generate random data within a range error.
Note	None.

3.2.3 asr_ecpki_buildprivkey

Prototype	int asr_ecpki_buildprivkey (const ECPKI_Domian_t *pDomian,const uint8_t *pPrivKeyIn, uint32_t privKeySizeInBytes, ECPKI_UserPrivKey_t *pUserPrivKey).
Function Description	Import the external input private key and Elliptic Curve parameters to an expected structure ECPKI_UserPrivKey_t before using them.
Input Parameters	● pDomain: Elliptic Curve parameters returned by asr_ecpki_getEcDomain. ● pPrivKeyIn: external input private key buffer. ● privKeySizeInBytes: size of input private key.
Output Parameters	● pUserPrivKey: output private key.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00802 -- Parameter pDomain is a NULL pointer. ● 0xF00809 -- Parameter pPrivKeyIn is a NULL pointer ● 0xF0080A -- Parameter pUserPrivKey is a NULL pointer. ● 0xF00895 -- Elliptic Curve parameters modSize or ordSize error.
Note	None.

3.2.4 asr_ecpki_exportprikey

Prototype	int asr_ecpki_exportprikey (ECPKI_UserPrivKey_t *pUserPrivKey, uint8_t *pExportPrivKey, uint32_t *pPrivKeySizeBytes).
Function Description	Export the private key.
Input Parameters	● pUserPrivKey: user private key structure generated by asr_ecpki_genkeypair or asr_ecpki_buildprivkey API. ● pPrivKeySizeBytes: export private key size.
Output Parameters	● pExportPrivKey: export private key buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00814 -- Parameter pUserPrivKey is a NULL pointer. ● 0xF00816 -- Parameter pExportPrivKey is a NULL pointer. ● 0xF00817 -- Parameter pPrivKeySizeBytes is a NULL pointer. ● 0xF00818 -- pPrivKeySizeBytes's value shall greater than ordSize. ● 0xF0081A -- The asr_ecpki_genkeypair or asr_ecpki_buildprivkey API should be called before calling this API.
Note	None.

3.2.5 asr_ecpki_buildpubkey

Prototype	int asr_ecpki_buildprivkey (const ECPKI_Domian_t *pDomian,const uint8_t *pPublKeyIn, uint32_t publKeySizeInBytes, USER_EC_PublKeyCheckMode_t checkMode, ECPKI_UserPublKey_t *pUserPublKey, ECPKI_BUILD_TempData_t *tempBuff).
Function Description	Import the external input public keys and Elliptic Curve domain to an expected structure ECPKI_UserPublKey_t before using them.
Input Parameters	● pDomain: including Elliptic Curve parameters returned by asr_ecpki_getEcDomain. ● PPublKeyIn: external input public key buffer. ● publKeySizeInBytes: size of external input public key. ● checkMode: public key check mode. ➢ Preliminary input parameters checking -- USER_CheckPointersAndSizeOnly ➢ Elliptic Curve public key is point on curve -- USER_ECpublKeyPartlyCheck ➢ Check EC_GenerateorOrder *PubKey = 0 -- USER_ECpublKeyFullCheck ● tempBuff: temporary buffer.
Output Parameters	● pUserPublKey: output public key.
Return Value	Return value and its meaning: ● 0x00 -- Success.

	● 0xF00802 -- Parameter pDomain is a NULL pointer. ● 0xF0080D -- Parameter pPublKeyIn is a NULL pointer. ● 0xF0080E -- Parameter pUserPublKey is a NULL pointer. ● 0xF00811 -- Parameter checkMode's value is error. ● 0xF00812 -- Parameter tempBuff is NULL and checkMode is USER_CheckPointersAndSizeOnly.
Note	None.

3.2.6 asr_ecpki_exportpubkey

Prototype	int asr_ecpki_exportpubkey (ECPKI_UserPublKey_t *pUserPublKey, ECPKI_PointCompression_t compression, uint8_t *pExportPublKey, uint32_t *pPublKeySizeBytes).
Function Description	Export public key.
Input Parameters	● pUserPublKey: user public key structure generated by asr_ecpki_genkeypair or asr_ecpki_buildpubkey API. ● compression: Elliptic Curve point operation. ➢ EC point compress -- EC_PointCompressed ➢ EC point uncompress -- EC_PointUncompressed ➢ EC point hybrid -- EC_PointHybrid ● pPublKeySizeBytes: output public key size.
Output Parameters	● pExportPublKey: output public key buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00802 -- Parameter pDomain is a NULL pointer. ● 0xF0080D -- Parameter pPublKeyIn is a NULL pointer. ● 0xF0080E -- Parameter pUserPublKey is a NULL pointer. ● 0xF00811 -- Parameter checkMode's value is error. ● 0xF00812 -- Parameter tempBuff is NULL and checkMode is USER_CheckPointersAndSizeOnly.
Note	None.

3.2.7 asr_ecdh_svdp_dh

Prototype	int asr_ecdh_svdp_dp (ECPKI_UserPublKey_t *PartnerPublKey_ptr, ECPKI_UserPrivKey_t *UserPrivKey_ptr, uint8_t *SharedSecretValue_ptr, uint32_t *SharedSecrValSize_ptr, ECDH_TempData_t *TempBuff_ptr).
Function Description	Get shared secret value from the input keys.
Input Parameters	● PartnerPublKey_ptr: the public key buffer. ● UserPrivKey_ptr: the private key buffer. ● TempBuff_ptr: a temporary buffer.
Output Parameters	● SharedSecretValue_ptr: shared secret value buffer. ● SharedSecrValSize_ptr: size of shared secret value.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00831 -- Parameter PartnerPublKey_ptr is a NULL pointer. ● 0xF00832 -- Public key is invalid, and the asr_ecpki_genkeypair or asr_ecpki_buildprivkey API should be called. ● 0xF00833 -- Parameter UserPrivKey_ptr is a NULL pointer. ● 0xF00834 -- Private key is invalid, and the asr_ecpki_genkeypair or asr_ecpki_buildprivkey API should be called. ● 0xF00835 -- Parameter SharedSecretValue_ptr is a NULL pointer. ● 0xF00836 -- Parameter SharedSecrValSize_ptr is a NULL pointer ● 0xF00837 -- Parameter TempBuff_ptr is a NULL pointer. ● 0xF00838 -- SharedSecrValSize_ptr's value is less than modSize. ● 0xF0083A -- Public key's domain is not equal to private key's.
Note	Public keys and private keys must be in the same domain.

3.2.8 asr_ecdsa_sign

Prototype	int asr_ecdsa_sign (RND_Context_t *pRndContext, ECDSA_SignUserContext_t *pSignUserContext, ECPKI_UserPrivKey_t *pSignerPrivKey, ECPKI_HASH_OpMode_t hashmod, uint8_t *pMessageDataIn, uint32_t messageSizeInBytes, uint8_t * pSignOut, uint32_t *pSignOutSize).
Function Description	ECC signature operation.
Input Parameters	● pRndContext: RND module's context. ● pSignUserContext: signature operation user context. ● pSignerPrivKey: private key buffer. ● hashmod: hash mode. ➢ HASH SHA1 mode -- ECPKI_HSAH_SHA1_mode ➢ HASH SHA256 mode -- ECPKI_HSAH_SHA256_mode ➢ HASH SHA512 mode -- ECPKI_HSAH_SHA512_mode ➢ After HASH SHA1 mode -- ECPKI_AFTER_HSAH_SHA1_mode ➢ After HASH SHA224 mode -- ECPKI_AFTER_HSAH_SHA224_mode ➢ After HASH SHA256 mode -- ECPKI_AFTER_HSAH_SHA256_mode ➢ After HASH SHA384 mode -- ECPKI_AFTER_HSAH_SHA384_mode ➢ After HASH SHA512 mode -- ECPKI_AFTER_HSAH_SHA512_mode ● pMessageDataIn: message data buffer. ● messageSizeInBytes: size of message data. ● pSignOutSize: size of the result of signature operation.
Output Parameters	● pSignOut: the result of signature operation.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00851 -- Parameter pSignUserContext is a NULL pointer. ● 0xF00852 -- Parameter pSignerPrivKey is a NULL pointer. ● 0xF00853 -- Hash mode is error. ● 0xF00854 -- Parameter pMessageDataIn is NULL and dataInSize is not 0. ● 0xF00855 -- Parameter dataInSize is greater than 1^29. ● 0xF00858 -- Private key is invalid, and asr_ecpki_genkeypair or asr_ecpki_buildprivkey API should be called. ● 0xF00860 -- Parameter pSignOut is a NULL pointer. ● 0xF00861 -- Parameter pSignOutSize is a NULL pointer. ● 0xF00865 -- Parameter pRndContext is a NULL pointer.
Note	None.

3.2.9 asr_ecdsa_verify

Prototype	int asr_ecdsa_verify (ECDSA_RND_Context_t *pVerifyUserContext, ECPKI_UserPublKey_t *pUserPublKey, ECPKI_HASH_OpMode_t hashMode, uint8_t *pSignatureIn, uint32_t SignatureSizeBytes, uint8_t * pMessageDataIn, uint32_t messageSizeInBytes).
Function Description	ECC verify operation.
Input Parameters	● pVerifyUserContext: user context of verify operation. ● pUserPublKey: user public key. ● hashMode: hash mode. ➢ HASH SHA1 mode -- ECPKI_HSAH_SHA1_mode ➢ HASH SHA256 mode -- ECPKI_HSAH_SHA256_mode ➢ HASH SHA512 mode -- ECPKI_HSAH_SHA512_mode ➢ After HASH SHA1 mode -- ECPKI_AFTER_HSAH_SHA1_mode ➢ After HASH SHA224 mode -- ECPKI_AFTER_HSAH_SHA224_mode ➢ After HASH SHA256 mode -- ECPKI_AFTER_HSAH_SHA256_mode ➢ After HASH SHA384 mode -- ECPKI_AFTER_HSAH_SHA384_mode ➢ After HASH SHA512 mode -- ECPKI_AFTER_HSAH_SHA512_mode ● pSignatureIn: input signature. ● SignatureSizeBytes: size of input signature. ● pMessageDataIn: input message data. ● messageSizeInBytes: size of input message data.
Output Parameters	● None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00871 -- Parameter pVerifyUserContext is a NULL pointer. ● 0xF00872 -- Parameter pUserPublKey is a NULL pointer. ● 0xF00873 -- Hash mode is error. ● 0xF00876 -- Parameter pSignatureIn is a NULL pointer. ● 0xF00880 -- Parameter pMessageDataIn is NULL and dataInSize is 0 Byte. ● 0xF00881 -- Parameter dataInSize is greater than or equal to 2^29. ● 0xF00883 -- Public key is invalid, and asr_ecpki_genkeypair or asr_ecpki_buildpubkey API should be called.
Note	None.

4. HASH

4.1 Functional Description

HASH algorithm is to make the input data of unfixed length to be the output data of fixed length. The fixed length depends on the hash mode.

4.2 APIs

4.2.1 asr_hash_init

Prototype	int asr_hash_init (HASHUserContext_t *ContextID_ptr, HASH_OperationMode_t OperationMode).
Function Description	Initialize HASH user context according to the operation mode.
Input Parameters	● ContextID_ptr: user context pointer. ● OperationMode: HASH operation mode. ➢ HASH SHA1 mode -- HASH_SHA1_mode ➢ HASH SHA224 mode -- HASH_SHA224_mode ➢ HASH SHA256 mode -- HASH_SHA256_mode ➢ HASH SHA512 mode -- HASH_SHA512_mode
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00200 -- Parameter ContextID_ptr is a NULL pointer. ● 0xF00201 -- Parameter OperationMode is error.
Note	None.

4.2.2 asr_hash_update

Prototype	int asr_hash_update (HASHUserContext_t *ContextID_ptr, uint8_t *DataIn_ptr, size_t DataInSize).
Function Description	It means to store data, and it can invoked several times.
Input Parameters	● ContextID_ptr: user context pointer. ● DataIn_ptr: input data pointer. ● DataInSize: size of input data.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00200 -- Parameter ContextID_ptr is a NULL pointer. ● 0xF00203 -- Parameter DataIn_ptr is NULL. ● 0xF00204 -- Parameter DataInSize is more than or equal to 2^29.
Note	None.

4.2.3 asr_hash_finish

Prototype	int asr_hash_finish (HASHUserContext_t *ContextID_ptr, HASH_Result_t HashResultBuff).
Function Description	HASH the input data to a fixed length data.
Input Parameters	ContextID_ptr: user context pointer.
Output Parameters	HashResultBuff: the output buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00200 -- Parameter ContextID_ptr is a NULL pointer. ● 0xF00205 -- Parameter HashResultBuff is a NULL pointer.
Note	None.

4.2.4 asr_hash

Prototype	int asr_hash (HASH_OperationMode_t OperationMode, uint8_t *DataIn_ptr, HASH_Result_t HashResultBuff).
Function Description	Asr_hash API integrates asr_hash_init, asr_hash_update and asr_hash_finish to itself.
Input Parameters	● OperationMode. ➢ HASH SHA1 mode -- HASH_SHA1_mode ➢ HASH SHA224 mode -- HASH_SHA224_mode ➢ HASH SHA256 mode -- HASH_SHA256_mode ➢ HASH SHA512 mode -- HASH_SHA512_mode ● DataIn_ptr: input data pointer.
Output Parameters	● HashResultBuff: the output buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success ● 0xF00200 -- Parameter ContextID_ptr is a NULL pointer. ● 0xF00200 -- Parameter is a NULL pointer. ● 0xF00201 -- Parameter OperationMode is error. ● 0xF00203 -- Parameter DataIn_ptr is NULL. ● 0xF00204 -- Parameter DataInSize is more than or equal to 2^29. ● 0xF00205 -- Parameter HashResultBuff is a NULL pointer.
Note	None.

5. HMAC

5.1 Functional Description

HMAC (HASH-based Message Authentication Code) algorithm uses HASH function and keys to make message authentication.

5.2 APIs

5.2.1 asr_HMAC_Init

Prototype	int asr_HMAC_Init (HMACUserContext_t *ContextID_ptr, HASH_OperationMode_t OperationMode, uint8_t *key_ptr, uint16_t keySize).
Function Description	Initialize HMAC user context with keys and hash operation mode.
Input Parameters	● ContextID_ptr: user context. ● OperationMode: hash operation mode. ➢ HASH SHA1 mode -- HASH_SHA1_mode ➢ HASH SHA224 mode -- HASH_SHA224_mode ➢ HASH SHA256 mode -- HASH_SHA256_mode ➢ HASH SHA512 mode -- HASH_SHA512_mode ● key_ptr: key pointer. ● keySize: size of key.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00300 -- Parameter ContextID_ptr is a NULL pointer. ● 0xF00301 -- Parameter OperationMode is illegal. ● 0xF00306 -- Parameter key_ptr is a NULL pointer. ● 0xF00307 -- Parameter keySize is 0 Byte.
Note	None.

5.2.2 asr_HMAC_Update

Prototype	int asr_HMAC_Update (HMACUserContext_t *ContextID_ptr, uint8_t *DataIn_ptr, size_t DataInSize).
Function Description	Initialize HMAC user context with keys and hash operation mode.
Input Parameters	● ContextID_ptr: user context. ● OperationMode: hash operation mode. ➢ HASH SHA1 mode -- HASH_SHA1_mode ➢ HASH SHA224 mode -- HASH_SHA224_mode ➢ HASH SHA256 mode -- HASH_SHA256_mode ➢ HASH SHA512 mode -- HASH_SHA512_mode ● key_ptr: key pointer. ● keySize: size of key.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00300 -- Parameter ContextID_ptr is a NULL pointer. ● 0xF00303 -- Parameter DataIn_ptr is a NULL pointer.
Note	None.

5.2.3 asr_HMAC_Finish

Prototype	int asr_HMAC_Finish (HMACUserContext_t *ContextID_ptr, HASH_Result_t HmacResultBuff).
Function Description	Calculating HMAC result.
Input Parameters	● ContextID_ptr: user context.
Output Parameters	● HmacResultBuff: HMAC result buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00300 -- Parameter ContextID_ptr is a NULL pointer. ● 0xF00305 -- Parameter HmacResultBuff is a NULL pointer.
Note	None.

5.2.4 asr_HMAC_Free

Prototype	int asr_HMAC_Free (HMACUserContext_t *ContextID_ptr).
Function Description	Freeing user context space.
Input Parameters	● ContextID_ptr : user context.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00300 -- Parameter ContextID_ptr is a NULL pointer.
Note	None.

5.2.5 asr_HMAC

Prototype	int asr_HMAC (HASH_OperationMode_t OperationMode, uint8_t *key_ptr, uint16_t keySize, uint8_t *DataIn_ptr, size_t DataSize, HASH_Result_t HmacResultBuf).
Function Description	Asr_HMAC API integrates asr_HMAC_Init, asr_HMAC_Update and asr_HMAC_Finish to itself.
Input Parameters	● OperationMode. ➢ HASH SHA1 mode -- HASH_SHA1_mode ➢ HASH SHA224 mode -- HASH_SHA224_mode ➢ HASH SHA256 mode -- HASH_SHA256_mode ➢ HASH SHA512 mode -- HASH_SHA512_mode ● key_ptr: input key pointer. ● DataIn_ptr: input data pointer. ● DataSize: size of input data.
Output Parameters	● HmacResultBuf: HMAC result buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00300 -- Parameter ContextID_ptr is a NULL pointer. ● 0xF00301 -- Parameter OperationMode is illegal. ● 0xF00303 -- Parameter DataIn_ptr is a NULL pointer. ● 0xF00305 -- Parameter HmacResultBuff is a NULL pointer. ● 0xF00306 -- Parameter key_ptr is a NULL pointer. ● 0xF00307 -- Parameter keySize is 0 Byte.
Note	None.

6. RND

6.1 Functional Description

RND module is to generate random data.

6.2 APIs

6.2.1 asr_RND_Instantiation

Prototype	int asr_RND_Instantiation (RND_Context_t *rndContext_ptr, RND_WorkBuff_t *rndWorkBuff_ptr).
Function Description	Instantiate the RND module and create new internal state (including seed) for RND module.
Input Parameters	● rndContext_ptr: context pointer of RND. ● rndWorkBuff_ptr: temporary workspace buffer.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00C20 -- Parameter rndWorkBuff_ptr is a NULL pointer. ● 0xF00C27 -- Parameter rndContext_ptr is a NULL pointer.
Note	None.

6.2.2 asr_RND_UnInstantiation

Prototype	int asr_RND_UnInstantiation (RND_Context_t *rndContext_ptr).
Function Description	Un-instantiation of RND module.
Input Parameters	● rndContext_ptr: context pointer of RND.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00C27 -- Parameter rndContext_ptr is a NULL pointer.
Note	None.

6.2.3 asr_RND_Reseeding

Prototype	int asr_RND_Reseeding (RND_Context_t *rndContext_ptr, RND_WorkBuff_t * rndWorkBuff_ptr).
Function Description	Reseeding for RND module, mixing additional entropy into the working state.
Input Parameters	● rndContext_ptr: context pointer of RND. ● rndWorkBuff_ptr: temporary workspace buffer.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00C20 -- Parameter rndWorkBuff_ptr is a NULL pointer. ● 0xF00C27 -- Parameter rndContext_ptr is a NULL pointer.
Note	None.

6.2.4 asr_RND_GenerateVector

Prototype	int asr_RND_GenerateVector (RND_State_t *rndState_ptr, uint16_t outSizeBytes, uint8_t *out_ptr).
Function Description	Generate random data.
Input Parameters	● rndState _ptr: state buffer pointer. ● outSizeBytes: size of random data.
Output Parameters	● out_ptr: random data buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00C00 -- Parameter out_ptr is a NULL pointer. ● 0xF00C0E -- asr_RND_Instantiation API should be called before this API. ● 0xF00C20 -- Parameter rndWorkBuff_ptr is a NULL pointer. ● 0xF00C27 -- Parameter rndContext_ptr is a NULL pointer.
Note	None.

6.2.5 asr_RND_GenerateVectorInRange

Prototype	int asr_RND_GenerateVectorInRange (RND_Context_t *rndContext_ptr, uint32_t rndSizeInBits, uint8_t *maxVect_ptr, uint8_t *rndVect_ptr).
Function Description	Generate a random vector in the range of (1, maxVect).
Input Parameters	● rndContext_ptr: context buffer pointer. ● rndSizeInBits: if maxVect_ptr is not given, then rndSizeInBits defining the exact size of generated random vector; If given, then it defines the size of the maxVect_ptr buffer. ● maxVect_ptr: max vector pointer.
Output Parameters	● rndVect_ptr: random data buffer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00C00 -- Parameter out_ptr is a NULL pointer. ● 0xF00C01 -- Generate random data within a range error. ● 0xF00C0E -- asr_RND_Instantiation API should be called earlier. ● 0xF00C14 -- ase_RND_SetGenerateVectorFunc should be called earlier. ● 0xF00C20 -- Parameter rndWorkBuff_ptr is a NULL pointer. ● 0xF00C27 -- Parameter rndContext_ptr is a NULL pointer. ● 0xF00C30 -- Parameter rndVect_ptr is a NULL pointer. ● 0xF00C31 -- Parameter rndSizeInBits is not in the range of (1,0xFFFF].
Note	None.

6.2.6 asr_RND_SetGenerateVectorFunc

Prototype	int asr_RND_SetGenerateVectorFunc (RND_Context_t *rndContext_ptr, RndGenerateVectWorkFunc_t rndGenerateVectFunc).
Function Description	Assign the address of a function that generates random data to a function pointer.
Input Parameters	● rndContext_ptr: context buffer of RND module. ● rndGenerateVectFunc: a function address parameter that is asr_RND_GenerateVector or asr_RND_GenerateVectorInRange.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00C14 -- Parameter rndGenerateVectFunc is a NULL pointer. ● 0xF00C27 -- Parameter rndContext_ptr is a NULL pointer.
Note	The prototype of the RndGenerateVectFunc_t parameter is Uint32_t (*RndGenerateVectFunc_t)(RND_State_t *rndState_ptr, uint16_t outSizeBytes, uint8_t *out_ptr).

6.2.7 asr_RND_AddAdditionalInput

Prototype	int asr_RND_AddAdditionalInput (RND_Context_t *rndContext_ptr, uint8_t *additionalInput_ptr, uint16_t additionalInputSize).
Function Description	Load the additional input data into RND context.
Input Parameters	● rndContext_ptr: context buffer of RND module. ● additionalInput_ptr: additional input data pointer. ● additionalInputSize: size of additional input data.
Output Parameters	None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00C03 -- additionalInput_ptr is NULL and additionalInputSize is not 0 Byte. ● 0xF00C04 -- additionalInputSize must be up to 48 Bytse and multiple of 4 Bytes. ● 0xF00C14 -- Parameter rndGenerateVectFunc is a NULL pointer. ● 0xF00C27 -- Parameter rndContext_ptr is a NULL pointer.
Note	None.

7. RSA

7.1 Functional Description

RSA algorithm is one of asymmetric encryption algorithms.

7.2 APIs

7.2.1 asr_rsa_keypair_gen

Prototype	int asr_rsa_keypair_gen (RND_Context_t *rndContext_ptr,uint8_t *pubExp_ptr, uint16_t pubExpSizeInBytes, uint32_t keySize,RSAUserPrivKey_t *userPrivKey_ptr, RSAUserPubKey_t *userPubKey_ptr, RSAKGData_t *keyGenData_ptr, RSAKGFipsContext_t *pFipsCtx).
Function Description	Generate RSA keys in non-CRT mode.
Input Parameters	● rndContext_ptr: RND context pointer. ● pubExp_ptr: public key exponent pointer, valid data are 0x03, 0x11 and 0x010001. ● pubExpSizeInBytes: size of Public key exponent in the unit of Bytes. ● keySize: support key sizes between 512 and 2048 bits in multiples of 256 bits long. ● keyGenData_ptr: a temporary key generation pointer. ● pFipsCtx: a context for FIPS certification.
Output Parameters	● userPrivKey_ptr: private keys pointer. ● userPubKey_ptr: public keys pointer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00400 -- Parameter keySize is not in the range of [512 2048] or keySize % 256 is not equal to 0. ● 0xF00402 -- Parameter pubExp_ptr is a NULL pointer. ● 0xF00403 -- Parameter userPubKey_ptr is a NULL pointer. ● 0xF00404 -- Parameter userPrivKey_ptr is a NULL pointer. ● 0xF00405 -- The pubExp_ptr's value is invalid. ● 0xF00406 -- The size of key e is 0 Byte or greater than 17 Bytes. ● 0xF00426 -- Parameter keyGenData_ptr is a NULL pointer.
Note	None.

7.2.2 asr_rsa_keypair_crt_gen

Prototype	int asr_rsa_keypair_crt_gen (RND_Context_t *rndContext_ptr,uint8_t *pubExp_ptr, uint16_t pubExpSizeInBytes, uint32_t keySize, RSAUserPrivKey_t *userPrivKey_ptr, RSAUserPubKey_t *userPubKey_ptr, RSAKGData_t *keyGenData_ptr, RSAKGFipsContext_t *pFipsCtx).
Function Description	Generate RSA keys in CRT mode.
Input Parameters	● rndContext_ptr: RND context pointer. ● pubExp_ptr: public key exponent pointer, valid data are 0x03,0x11 and 0x010001. ● pubExpSizeInBytes: size of Public key exponent in the unit of bytes. ● keySize: support key sizes between 512 and 2048 bits in multiples of 256 bits long. ● keyGenData_ptr: a temporary key generation pointer. ● pFipsCtx: a context for FIPS certification.
Output Parameters	● userPrivKey_ptr: private keys pointer. ● userPubKey_ptr: public keys pointer.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00400 -- Parameter keySize is not in the range of [512 2048] or keySize % 256 is not equal to 0. ● 0xF00402 -- Parameter pubExp_ptr is a NULL pointer. ● 0xF00403 -- Parameter userPubKey_ptr is a NULL pointer. ● 0xF00404 -- Parameter userPrivKey_ptr is a NULL pointer. ● 0xF00405 -- The pubExp_ptr's value is invalid. ● 0xF00406 -- The size of key e is 0 Byte or greater than 17 Bytes. ● 0xF00426 -- Parameter keyGenData_ptr is a NULL pointer.
Note	None.

7.2.3 asr_rsa_build_pubkey

Prototype	int asr_rsa_build_pubkey (RSAUserPubKey_t *UserPubKey_ptr, uint8_t *Exponent_ptr, uint16_t ExponentSize, uint8_t *Modulus_ptr, uint16_t ModulusSize).
Function Description	Import RSA public key (n,e) into the structure RSAUserPubKey_t.
Input Parameters	● Exponent_ptr: public key e byte stream in big endian. ● ExponentSize: size of public key e. ● Modulus_ptr: public key n byte stream in big endian. ● ModulusSize: size of public key n.
Output Parameters	● UserPubKey_ptr: private key structure.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00400 -- Size of key n is invalid. ● 0xF00401 -- Parameter Modulus_ptr is a NULL pointer. ● 0xF00402 -- Parameter Exponent_ptr is a NULL pointer. ● 0xF00403 -- Parameter UserPubKey_ptr is a NULL pointer. ● 0xF00405 -- The value of key e is less than 3 Bytes. ● 0xF00406 -- Size of key e is 0 Byte. ● 0xF004015 -- The MSB of key n's first Byte is odd.
Note	None.

7.2.4 asr_rsa_build_prikey

Prototype	int asr_rsa_build_prikey (RSAUserPrivKey_t *UserPrivKey_ptr, uint8_t *PrivExponent_ptr, uint16_t PrivExponentSize, uint8_t *PubExponent_ptr, uint16_t PubExponentSize,uint8_t *Modulus_ptr, uint16_t ModulusSize).
Function Description	Import RSA key (n,e,d) into the structure RSAUserPrivKey_t.
Input Parameters	● PrivExponent_ptr: key d's byte stream in big endian. ● PrivExponentSize: size of key d. ● PubExponent_ptr: key e's byte stream in big endian. ● PubExponentSize: size of key e. ● Modulus_ptr: key n's byte stream in big endian. ● ModulusSize: size of key n.
Output Parameters	● UserPrivKey_ptr: private key structure.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00400 -- ModulusSize is greater than 256 Bytes. ● 0xF00401 -- Parameter Modulus_ptr is a NULL pointer. ● 0xF00402 -- Parameter PrivExponent_ptr is a NULL pointer. ● 0xF00404 -- Parameter UserPrivKey_ptr is a NULL pointer. ● 0xF00405 -- The first byte of key d is less than 1 Byte. ● 0xF00406 -- PrivExponentSize or PubExponentSize is greater than 256 Bytes. ● 0xF00415 -- The MSB of key n's first byte is 1 Byte.
Note	None.

7.2.5 asr_rsa_build_prikey_crt

Prototype	int asr_rsa_build_prikey_key (RSAUserPrivKey_t *UserPrivKey_ptr, uint8_t *P_ptr, uint16_t PSize, uint8_t *Q_ptr, uint16_t QSize, uint8_t *dP_ptr, uint16_t dPSize, uint8_t *dQ_ptr, uint16_t dQSize, uint8_t *qInv_ptr, uint16_t qInvSize).
Function Description	Import RSA key (n,e,d) into the structure RSAUserPrivKey_t.
Input Parameters	● P_ptr: key p's byte stream in big endian. ● PSize: size of key p. ● Q_ptr: key q's byte stream in big endian. ● QSize: size of key q. ● dP_ptr: key dp's byte stream in big endian. ● dpSize: size of key dp. ● dQ_ptr: key dq's byte stream in big endian. ● dqSize: size of key dq. ● qInv_ptr: key qInv's byte stream in big endian. ● qInvSize : size of key qInv.
Output Parameters	● UserPrivKey_ptr: private key structure in CRT mode.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00400 -- The size of key n = p*q is wrong. ● 0xF00404 -- Parameter UserPrivKey_ptr is a NULL pointer. ● 0xF00407 -- Parameter P_ptr is a NULL pointer. ● 0xF00408 -- Parameter Q_ptr is a NULL pointer. ● 0xF00409 -- Parameter dP_ptr is a NULL pointer. ● 0xF0040A -- Parameter dQ_ptr is a NULL pointer. ● 0xF0040B -- Parameter qInv_ptr is a NULL pointer. ● 0xF0043A -- PSize, QSize, qInvSize, dPSize or dQSize is wrong.
Note	None.

7.2.6 asr_rsa_get_pubkey

Prototype	int asr_rsa_get_pubkey (RSAUserPubKey_t *UserPubKey_ptr, uint8_t *Exponent_ptr, uint16_t *ExponentSIze_ptr, uint8_t *Modulus_ptr, uint16_t *ModulusSize_ptr).
Function Description	Get public key (n,e) byte stream.
Input Parameters	● UserPubKey_ptr: public key structure buffer.
Output Parameters	● Exponent_ptr: key e's buffer. ● ExponentSIze_ptr: size of key e. ● Modulus_ptr: key n's buffer. ● ModulusSize_ptr: size of key n.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00802 -- Parameter Exponent_ptr or Modulus_ptr is a NULL pointer. ● 0xF00806 -- Size of key e is wrong. ● 0xF00803 -- Parameter UserPubKey_ptr is a NULL pointer. ● 0xF0081B -- Public key is not generated or imported externally. ● 0xF0082A -- Parameter ModulusSize_ptr is a NULL pointer. ● 0xF0082B -- Parameter ExponentSize is a NULL pointer.
Note	None.

7.2.7 asr_rsa_sign

Prototype	int asr_rsa_sign (RND_Context_t *rndContext_ptr, RSAPrivUserContext_t *UserContext_ptr, RSAUserPrivKey_t *UserPrivKey_ptr, RSA_HASH_OpMode_t rsaHashMode, PKCS1_MSG_t MGF, uint16_t SaltLen, uint8_t *DataIn_ptr, uint32_t DataInSize, uint8_t *Output_ptr, uint16_t *OutputSize_ptr, PKCS1_version PKCS1_ver).
Function Description	RSA signature operation.
Input Parameters	● rndContext_ptr: context buffer of RND module. ● UserContext_ptr: temporary context for internal use. ● UserPrivKey_ptr: private key structure. ● rsaHashMode: hash mode. ➢ SHA1 mode -- RSA_HASH_SHA1_mode ➢ SHA224 mode -- RSA_HASH_SHA224_mode ➢ SHA256 mode -- RSA_HASH_SHA256_mode ➢ SHA512 mode -- RSA_HASH_SHA512_mode ➢ After MD5 mode -- RSA_After_MD5_mode ➢ After SHA1 mode -- RSA_After_SHA1_mode ➢ After SHA224 mode -- RSA_After_SHA224_mode ➢ After SHA256 mode -- RSA_After_SHA256_mode ➢ After SHA384 mode -- RSA_After_SHA384_mode

	➢ After SHA512 mode -- RSA_After_SHA512_mode ● MSF: PKCS1 mode. ➢ PKCS1 MGF1 -- PKCS1_MGF1 ➢ PKCS1_NO_MGF -- PKCS1_NO_MGF ● SaltLen: length of salt. ● DataIn_ptr: input data to signature. ● PKCS1_ver: PKCS1 version. ➢ PKCS1 ver15 -- PKCS1_VER15 ➢ PKCS1 ver21 -- PKCS1_VER21
Output Parameters	● Output_ptr: signature data. ● OutputSize_ptr: size of output data.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00404 -- Parameter UserPrivKey_ptr is a NULL pointer. ● 0xF00412 -- Parameter DataIn_ptr is a NULL pointer. ● 0xF00413 -- Parameter DataInSize is greater than 2^29. ● 0xF00416 -- Parameter UserContext_ptr is a NULL pointer. ● 0xF00417 -- Parameter PKCS1_ver is PKCS1_VER21 and rsaHashMode is RSA_HASH_MD5_mode or rsaHashMode is wrong. ● 0xF00418 -- Parameter MGF is wrong. ● 0xF00419 -- Parameter PKCS1_ver is wrong. ● 0xF0041A -- Private key is not generated or imported externally. ● 0xF0041D -- Parameter Output_ptr is a NULL pointer. ● 0xF0041F -- Parameter OutputSize_ptr is a NULL pointer. ● 0xF00429 -- The size of output data is wrong.
Note	None.

7.2.8 asr_rsa_verify

Prototype	int asr_rsa_verify (RSAPubUserContext_t *UserContext_ptr, RSAUserPubKey_t *UserPubKey_ptr, RSA_HASH_OpMode_t rsaHashMode, PKCS1_MGF_t MGF, uint16_t SaltLen, uint8_t *DataIn_ptr, uint32_t DataInSize, uint8_t *Sig_ptr, PKCS1_version PKCS1_ver).
Function Description	RSA verify operation.
Input Parameters	● UserContext_ptr: temporary context for internal use. ● UserPubKey_ptr: public key structure. ● rsaHashMode: hash mode. ➢ SHA1 mode -- RSA_HASH_SHA1_mode ➢ SHA224 mode -- RSA_HASH_SHA224_mode ➢ SHA256 mode -- RSA_HASH_SHA256_mode ➢ SHA512 mode -- RSA_HASH_SHA512_mode ➢ After MD5 mode -- RSA_After_MD5_mode ➢ After SHA1 mode -- RSA_After_SHA1_mode ➢ After SHA224 mode -- RSA_After_SHA224_mode ➢ After SHA256 mode -- RSA_After_SHA256_mode ➢ After SHA384 mode -- RSA_After_SHA384_mode ➢ After SHA512 mode -- RSA_After_SHA512_mode ● MGF: PKCS1 mode. ➢ PKCS1 MGF1 -- PKCS1_MGF1 ➢ PKCS1_NO_MGF -- PKCS1_NO_MGF ● SaltLen: length of salt. ● DataIn_ptr: input data to verify. ● DataInSize: size of input data. ● Sig_ptr: signature data buffer. ● PKCS1_ver: PKCS1 version. ➢ PKCS1 ver15 -- PKCS1_VER15 ➢ PKCS1 ver21 -- PKCS1_VER21
Output Parameters	● None.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00403 -- Parameter UserPubKey_ptr is a NULL pointer. ● 0xF00412 -- Parameter DataIn_ptr is a NULL pointer. ● 0xF00413 -- Parameter DataInSize is greater or equal to 2^29. ● 0xF00416 -- Parameter UserContext_ptr is a NULL pointer. ● 0xF00417 -- Parameter PKCS1_ver is PKCS1_VER21 and rsaHashMode is RSA_HASH_MD5_mode or rsaHashMode is wrong. ● 0xF00418 -- Parameter MGF is wrong. ● 0xF00419 -- Parameter PKCS1_ver is wrong. ● 0xF0042C -- Parameter Sig_ptr is a NULL pointer.
Note	None

7.2.9 asr_rsa_encrypt

Prototype	int asr_rsa_encrypt (RND_Context_t *rndContext_ptr, RSAUserPubKey_t *UserPubKey_ptr, RSAPrimeData_t *PrimeData_ptr, RSA_HASH_OpMode_t hashFunc, uint8_t *L, uint16_t Llen, PKCS1_MGF_t MGF, uint8_t *DataIn_ptr, uint6_t DataInSize, uint8_t *Output_ptr, PKCS1_version PKCS1_ver).
Function Description	RSA encrypt operation.
Input Parameters	● rndContext_ptr: context of RND module. ● UserPubKey_ptr: private key structure. ● PrimeData_ptr: prime data buffer. ● hashFunc: hash mode. ➢ SHA1 mode -- RSA_HASH_SHA1_mode ➢ SHA224 mode -- RSA_HASH_SHA224_mode ➢ SHA256 mode -- RSA_HASH_SHA256_mode ➢ SHA512 mode -- RSA_HASH_SHA512_mode ➢ NO Hash mode -- RSA_HASH_NO_HASH_mode ● L: label input buffer, relative to PKCS1_VER21. ● Llen: length of label input buffer. ● MGF: PKCS1 mode. ➢ PKCS1 MGF1 -- PKCS1_MGF1 ➢ PKCS1_NO_MGF -- PKCS1_NO_MGF ● DataIn_ptr: plaintext data. ● DataInSize: size of plaintext data. ● PKCS1_ver: PKCS1 version. ➢ PKCS1 ver15 -- PKCS1_VER15 ➢ PKCS1 ver21 -- PKCS1_VER21
Output Parameters	● Output_ptr: ciphertext data.
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00403 -- Parameter UserPubKey_ptr is a NULL pointer. ● 0xF00412 -- Parameter DataIn_ptr is a NULL pointer. ● 0xF00417 -- Parameter hashFunc is wrong. ● 0xF0041D -- Parameter Output_ptr is wrong. ● 0xF00418 -- Parameter MGF is wrong. ● 0xF00419 -- Parameter PKCS1_ver is wrong. ● 0xF0041B -- Public key should be generated or imported. ● 0xF00427 -- Parameter PrimeData_ptr is a NULL pointer.
Note	None.

7.2.10 asr_rsa_decrypt

Prototype	int asr_rsa_decrypt (RSAUserPrivate_t *UserPrivKey_ptr, RSAPrimeData_t RSAPrimeData_ptr, RSA_HASH_OpMode_t hashFunc, uint8_t * L, uint16_t Llen, PKCS1_MGF_t MGF, uint8_t *DataIn_ptr,uint16_t Dataput_ptr, uint8_t *Output_ptr, uin16_t *OutputSize_ptr, uit16_t *OutputSize_ptr,PKCS1_version PKCS1_ver).
Function Description	RSA decrypt operation.
Input Parameters	● UserPrivKey_ptr: private key buffer. ● RSAPrimeData_ptr: a temporary. ● hashFunc: hash mode. ➢ SHA1 mode -- RSA_HASH_SHA1_mode ➢ SHA224 mode -- RSA_HASH_SHA224_mode ➢ SHA256 mode -- RSA_HASH_SHA256_mode ➢ SHA512 mode -- RSA_HASH_SHA512_mode ➢ NO Hash mode -- RSA_HASH_NO_HASH_mode ● L: label input buffer, relative to PKCS1_VER21. ● Llen: length of label input buffer. ● MGF: MGF: PKCS1 mode. ➢ PKCS1 MGF1 -- PKCS1_MGF1 ➢ PKCS1_NO_MGF -- PKCS1_NO_MGF ● DataIn_ptr: data to be decrypted. ● DataInSize: size of data to be decrypted. ● PKCS1_ver: PKCS1 version. ➢ PKCS1 ver15 -- PKCS1_VER15 ➢ PKCS1 ver21 -- PKCS1_VER21
Output Parameters	● Output_ptr: output buffer. ● OutpSize: size of output buffer .
Return Value	Return value and its meaning: ● 0x00 -- Success. ● 0xF00404 -- Parameter UserPriKey is a NULL pointer. ● 0xF00412 -- Parameter DataIn_ptr is invalid. ● 0xF00413 -- DataInSize is not equal to key n size. ● 0xF00417 -- Parameter hashFunc is invalid. ● 0xF00418 -- MGF is invalid. ● 0xF00419 -- PKCS1_ver is invalid. ● 0xF0041D -- Parameter Output_ptr is a NULL pointer. ● 0xF00427 -- Parameter PrimeData_ptr is a NULL pointer. ● 0xF0048A -- Parameter OutputSize_ptr is a NULL pointer.
Note	None.

