USER'S GUIDE

Apollo5 Family Security

Ultra-Low Power Apollo SoC Family

A-SOCAP5-UGGA04EN v1.0

Legal Information and Disclaimers

AMBIQ MICRO INTENDS FOR THE CONTENT CONTAINED IN THE DOCUMENT TO BE ACCURATE AND RELIABLE. THIS CONTENT MAY, HOWEVER, CONTAIN TECHNICAL INACCURACIES, TYPOGRAPHICAL ERRORS OR OTHER MISTAKES. AMBIQ MICRO MAY MAKE CORRECTIONS OR OTHER CHANGES TO THIS CONTENT AT ANY TIME. AMBIQ MICRO AND ITS SUPPLIERS RESERVE THE RIGHT TO MAKE CORRECTIONS, MODIFICATIONS, ENHANCEMENTS, IMPROVEMENTS AND OTHER CHANGES TO ITS PRODUCTS, PROGRAMS AND SERVICES AT ANY TIME OR TO DISCONTINUE ANY PRODUCTS, PROGRAMS, OR SERVICES WITHOUT NOTICE.

THE CONTENT IN THIS DOCUMENT IS PROVIDED "AS IS". AMBIQ MICRO AND ITS RESPECTIVE SUPPLIERS MAKE NO REPRESENTATIONS ABOUT THE SUITABILITY OF THIS CONTENT FOR ANY PURPOSE AND DISCLAIM ALL WARRANTIES AND CONDITIONS WITH REGARD TO THIS CONTENT, INCLUDING BUT NOT LIMITED TO, ALL IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHT.

AMBIQ MICRO DOES NOT WARRANT OR REPRESENT THAT ANY LICENSE, EITHER EXPRESS OR IMPLIED, IS GRANTED UNDER ANY PATENT RIGHT, COPYRIGHT, MASK WORK RIGHT, OR OTHER INTELLECTUAL PROPERTY RIGHT OF AMBIQ MICRO COVERING OR RELATING TO THIS CONTENT OR ANY COMBINATION, MACHINE, OR PROCESS TO WHICH THIS CONTENT RELATE OR WITH WHICH THIS CONTENT MAY BE USED.

USE OF THE INFORMATION IN THIS DOCUMENT MAY REQUIRE A LICENSE FROM A THIRD PARTY UNDER THE PATENTS OR OTHER INTELLECTUAL PROPERTY OF THAT THIRD PARTY, OR A LICENSE FROM AMBIQ MICRO UNDER THE PATENTS OR OTHER INTELLECTUAL PROPERTY OF AMBIQ MICRO.

INFORMATION IN THIS DOCUMENT IS PROVIDED SOLELY TO ENABLE SYSTEM AND SOFTWARE IMPLEMENTERS TO USE AMBIQ MICRO PRODUCTS. THERE ARE NO EXPRESS OR IMPLIED COPYRIGHT LICENSES GRANTED HEREUNDER TO DESIGN OR FABRICATE ANY INTEGRATED CIRCUITS OR INTEGRATED CIRCUITS BASED ON THE INFORMATION IN THIS DOCUMENT. AMBIQ MICRO RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS HEREIN. AMBIQ MICRO MAKES NO WARRANTY, REPRESENTATION OR GUARANTEE REGARDING THE SUITABILITY OF ITS PRODUCTS FOR ANY PARTICULAR PURPOSE, NOR DOES AMBIQ MICRO ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT OR CIRCUIT, AND SPECIFICALLY DISCLAIMS ANY AND ALL LIABILITY, INCLUDING WITHOUT LIMITATION CONSEQUENTIAL OR INCIDENTAL DAMAGES. “TYPICAL” PARAMETERS WHICH MAY BE PROVIDED IN AMBIQ MICRO DATA SHEETS AND/OR SPECIFICATIONS CAN AND DO VARY IN DIFFERENT APPLICATIONS AND ACTUAL PERFORMANCE MAY VARY OVER TIME. ALL OPERATING PARAMETERS, INCLUDING “TYPICALS” MUST BE VALIDATED FOR EACH CUSTOMER APPLICATION BY CUSTOMER’S TECHNICAL EXPERTS. AMBIQ MICRO DOES NOT CONVEY ANY LICENSE UNDER NEITHER ITS PATENT RIGHTS NOR THE RIGHTS OF OTHERS. AMBIQ MICRO PRODUCTS ARE NOT DESIGNED, INTENDED, OR AUTHORIZED FOR USE AS COMPONENTS IN SYSTEMS INTENDED FOR SURGICAL IMPLANT INTO THE BODY, OR OTHER APPLICATIONS INTENDED TO SUPPORT OR SUSTAIN LIFE, OR FOR ANY OTHER APPLICATION IN WHICH THE FAILURE OF THE AMBIQ MICRO PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR. SHOULD BUYER PURCHASE OR USE AMBIQ MICRO PRODUCTS FOR ANY SUCH UNINTENDED OR UNAUTHORIZED APPLICATION, BUYER SHALL INDEMNIFY AND HOLD AMBIQ MICRO AND ITS OFFICERS, EMPLOYEES, SUBSIDIARIES, AFFILIATES, AND DISTRIBUTORS HARMLESS AGAINST ALL CLAIMS, COSTS, DAMAGES, AND EXPENSES, AND REASONABLE ATTORNEY FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PERSONAL INJURY OR DEATH ASSOCIATED WITH SUCH UNINTENDED OR UNAUTHORIZED USE, EVEN IF SUCH CLAIM ALLEGES THAT AMBIQ MICRO WAS NEGLIGENT REGARDING THE DESIGN OR MANUFACTURE OF THE PART.

Revision History

Revision	Date	Description
1.0	May 2025	Initial release

Reference Documents

Document ID	Description
A-SOCAP5-UGGA03EN	Apollo5 Family OEM Provisioning and Tools User's Guide
A-SOCAP5-UGGA02EN	Apollo5 Family Secure Update User's Guide
A-SOCAP5-UGGA01EN	Apollo5 MRAM Recovery Guide

Table of Contents

List of Tables

List of Figures

SECTION 1 Terminology

This section defines the abbreviation and terminology used in this document.

Table 1-1: Terminology

Abbreviation	Definition	Meaning
AES	Advanced Encryption Standard	
	BootROM	This is a ROM program that originally boots on reset and provides the initial immutable RoT. In the Apollo5 devices, the bootrom is referred to as the SBR, as it also enforces the first steps in the Secure Boot process in the ROM.
	Certificate Chain	A dependent sequence of certificates that together can be used validate content in the SoC memory. The certificate chain is composed of key certificates and a content certificate. The advantage of a chain is that some of the links can change without installing a completely new chain.
CM LCS	Chip Manufacturing LCS	Ambiq acting with the fabrication, packaging, and test house partners (e.g., ICV) is considered the Chip Manufacturer. In this LCS mode, the SoC is mostly open and ready for initial provisioning with Ambiq security assets.
	Content Certificate	A certificate used to identify and validate software components.
	Cryptocell-312	CryptoCell 312 (CRYPTOCELL) is a security subsystem providing root of trust (RoT) and cryptographic services for a device.
DCU	Debug Control Unit	CryptoCell-312 mechanism to control the debug mechanism and feature enablement.
DM LCS	Device Manufacturing LCS	The LCS mode when the OEM (Ambiq's customers) receives the Apollo5 device. In this LCS, the customer provisions their security policy and assets.
HBK	Hash of the Public Key	Ambiq uses the Dual scheme: ▪ Hbk0: A 128-bit truncated SHA-256 digest of the ICV PubKB0. ▪ Hbk1: A 128-bit truncated SHA-256 digest of the OEM PubKB1.
HMAC	Hashed Message Authentication Code	A cryptographic authentication technique that uses a hash function and a secret key

Table 1-1: Terminology *(Continued)*

Abbreviation	Definition	Meaning
HUK	Hardware Unique Key	This is a 256-bit SoC-unique AES key that is used as the root for deriving all SoC-specific keys.
ICV	Integrated Chip Vendor	ICV (e.g., CM) is Ambiq acting with the fabrication, packaging, and test house partners.
INFOC	InfoSpace Region C	This is an area of memory inside the SoC that is OTP (One Time Programmable) where the OEM provisions their "keys" and security policies and other related security assets.
INFO0	InfoSpace Region #0	This is an area of memory inside the SoC that is designed to be programmed by the OEM and hold customer configurations and customer define attributes. There are two instances of INFO0, OTP (one time programmable) and MRAM (reconfigurable).
INFO1	InfoSpace Region #1	This is an area of memory inside the SoC that is design to be programmed by Ambiq to hold SoC trims and security assets with a portion being OEM visible.
IPT	ICV Provisioning Tool	A tool used by Ambiq’s internal teams (software, validation, production test) to provision the secure assets of the SoC when in CM mode.
Kce	OEM Code Encryption Key	128-bit AES key that is used to decrypt OEM software images as part of the Secure Boot process.
Kceicv	ICV Code Encryption Key	128-bit AES key that is used to decrypt Ambiq software images as part of the Secure Boot process.
Kcp	OEM Provisioning Master Key	128-bit AES key that is used for the OEM asset provisioning flow.
	Key Certificate	A certificate used to validate the next certificate in a chain.
Kpicv	ICV Provisioning Master Key	128-bit AES key that is used for the ICV asset provisioning flow.
Krtl	Platform key	An ICV key placed in the RTL. It is used for provisioning during production life-cycle states.
LCS	Life Cycle State	The state of the SoC that determines the level of security and access to features. CM (Chip Manufacturer), DM (Device Manufacturer), SE (Secure), or RMA (Return Merchandise Authorization)..
OEM	Original Equipment Manufacturer	These are typically Ambiq’s direct customers and consumers of the Apollo5 SoCs (that receive the Apollo5 in DM-LCS state).
OPT	OEM Provisioning Tool	A set of example scripts for the purpose of providing an example of the OEM provisioning flow. It is envisioned that these scripts will be adopted and/or modified by the OEM into their production flow.
OTP	One Time Programmable memory	This is an area of memory in the Apollo5 that can be programmed only once during provisioning. There are three areas of OTP available, INFOC-OTP, INFO0-OTP and INFO1-OTP. INFO0 can be customer altered to be sourced from OTP or MRAM. INFO1 will always be set to OTP from the factory and cannot be changed. INFOC is always (only) OTP.
PKI	Public Key Infrastructure	A set of roles, policies, hardware, software and procedures needed to create, manage, distribute, use, store and revoke digital certificates and manage public-key encryption.

Table 1-1: Terminology *(Continued)*

Abbreviation	Definition	Meaning
RMA LCS	Return Merchandise Authorization	This is the process of returning a faulty SoC. The RMA LCS is a special state that does not allow recovery of the SoC. As the RMA LCS is entered, the security assets for the ICV and OEM are erased.
RoT	Root of Trust	This is the original source that can always be trusted within a cryptographic system.
RSA	Rivest–Shamir–Adleman	An algorithm used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of the keys is public and can be given to anyone.
SBL	Secure Boot Loader	The second stage of secure boot of the Apollo5. The SBL is validated as part of the Secure boot rom (SBR), and allows for secure loading and updating of OEM assets (and of itself). When Secure boot is enabled it will validate the OEM main image before passing control to it.
SBR	Secure BootROM	The first stage of secure boot processing built into the Bootrom of the Apollo5. It ensures the security of the device by validating the next software before its allowed to be run.
SE LCS	Secure Enable LCS	This is the state when the SoC is fully secured and content is authenticated and updated according to the established OEM security policy. Debugging is disabled by default. This is the proper state when the SoC is “in the field”.
Secure OTA	Secure Over the Air programming	This is the process of authenticating and installing a “blob” of data that has been delivered to the SBL. This blob could be received over any radio (air) or wired interface (I^2C, SPI, SWO, UART).
Wired Update		This is the process of receiving update images via the wired protocol that sends a sequence of packets over a wired interface (SPI or UART).

SECTION

2

Introduction

NOTES:

- This document assumes "public side crypto enabled devices".
- This document applies to the Apollo5 family of SoCs. Any reference to Apollo5 pertains to any device in the family unless explicitly stated otherwise.
- Each device in the Apollo5 Family has its own register documentation within AmbiqSuite, for example **/docs/registers/apollo5xx** and **/mcu/apollo5xx** will be specific to the **5xx** device.
- The Apollo510 device utilizes the **/tools/Apollo510_scripts**/ directory.

The security features of the Apollo5 system on chip (SoC) enables a trusted execution environment for resource constrained wearable and IoT devices. The Apollo5 supports the Arm® Platform Security Architecture (PSA) and provide robust system level security leveraging Ambiq's secureSPOT® technology. The Apollo5 security includes Arm Cryptocell hardware cryptographic acceleration, INFOC One-Time Programmable memory (INFOC-OTP), secure boot firmware in ROM to ensure a complete end-to-end secure environment for customer applications. The Apollo5 establishes and maintains a Root-of-Trust (RoT) throughout the SoC boot, reset, and firmware upgrade flows.

The specific security features supported by the Apollo5 SoC include:

- Secure Boot using Root-of-Trust in OTP
- Secure Over-the-Air (OTA) Updates
- Secure Wired Updates over SPI, or UART
- Secure Key Storage in INFOC-OTP
- Support for Secure and Non-secure modes of operation
- Lifecycle States (LCS) management

- Secure Debug using Secure Debug Certificates which provide selective debug controls and/or transition into RMA LCS.
- Support for Certificate Chain based authentication for customer image(s) before they are run, which are verified using PKI rooted in on chip RoT.
- Hardware Crypto Acceleration and True Random Number Generator (TRNG)
- CRC32
- Tools for asset provisioning during manufacturing
- Support for customer specific secondary boot loader (to be executed after Ambiq’s Secure Bootloader)

This document provides an overview to new customers on how to work with the secureSPOT features of the Apollo5 SoCs. This document assumes the customer will supplement the information provided herein with the Reference Documents noted earlier. Information presented in those documents is key to understanding this document in depth.

The Apollo5 secureSPOT includes cryptographic hardware acceleration, one-time programmable memory, customer trimming, Secure BootROM, Secure Bootloader, runtime security API, and provisioning tools designed to operate as a complete package to ensure protection of both Ambiq and OEM’s assets.

The first part of this document focuses on the Apollo5 Security infrastructure overview, and second part on configuring, provisioning and usage.

The document will cover the following important topics:

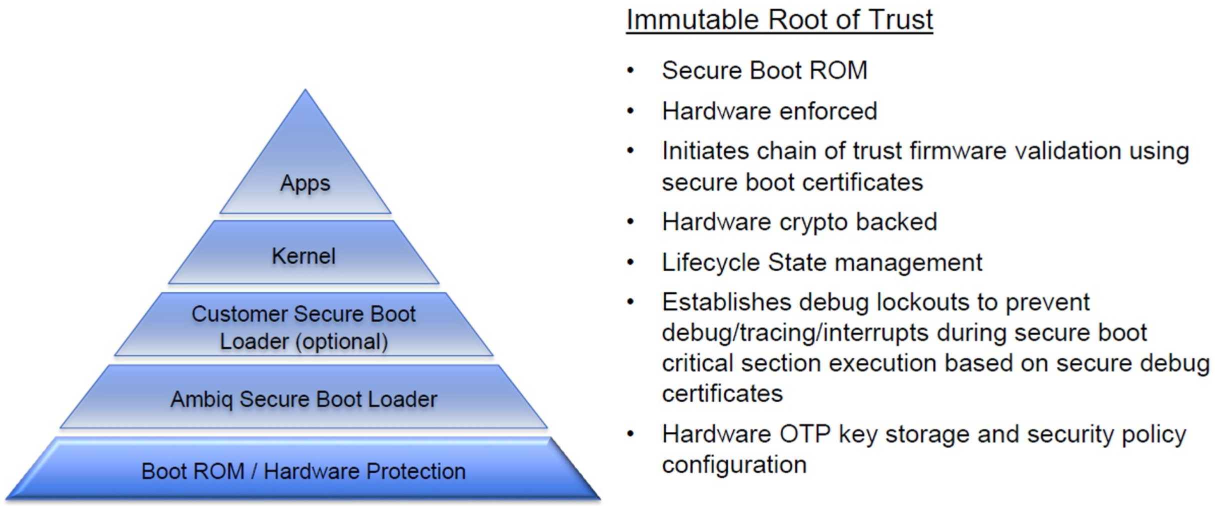
- Apollo5 Security infrastructure
  - Apollo5 Secure Boot overview
  - Security Life Cycle States (LCS)
  - Secure Boot Mode
  - Debug Support
  - Run Time Security Services
- Configuration, Provisioning & Usage
  - Default SoC Configuration
  - INFOC-OTP Configuration
  - Configuring OEM InfoSpace (INFO0)
    - (both MRAM and OTP areas)
  - Configuring Secure Boot Mode
  - Transition to Secure LCS
  - Transition to RMA LCS
  - Image Updates

SECTION 3

Apollo5 Secure Boot Overview

At a high level, the Apollo5 Secure Boot architecture and features are modeled in Figure 3-1.

Figure 3-1: Apollo5 Secure Boot Architecture


Apps
Kernel
Customer Secure Boot Loader (optional)
Ambiq Secure Boot Loader
Boot ROM / Hardware Protection

Immutable Root of Trust

- Secure Boot ROM
- Hardware enforced
- Initiates chain of trust firmware validation using secure boot certificates
- Hardware crypto backed
- Lifecycle State management
- Establishes debug lockouts to prevent debug/tracing/interrupts during secure boot critical section execution based on secure debug certificates
- Hardware OTP key storage and security policy configuration

The base of the security model is the Root-of-Trust in INFOC-OTP memory which is provisioned during chip and device manufacturing. Upon a power-cycle, the Secure BootROM validates the SoC trims and security settings. The Secure BootROM provides for CM and DM state provisioning of the security assets and policies in the INFOC-OTP memory. The SBR also performs Certificate Chain based verification of the Secure Bootloader (SBL) and processing of Secure Debug certificates. The SBL includes support for OTA firmware updates, self-updates, wired updates, certificate chain updates, key revocation, trim patch updates, and certificate chain based authentication of optional OEM secondary bootloaders or main images.

3.1 Secure Boot Functional Components

3.1.1 Secure BootROM (Immutable)

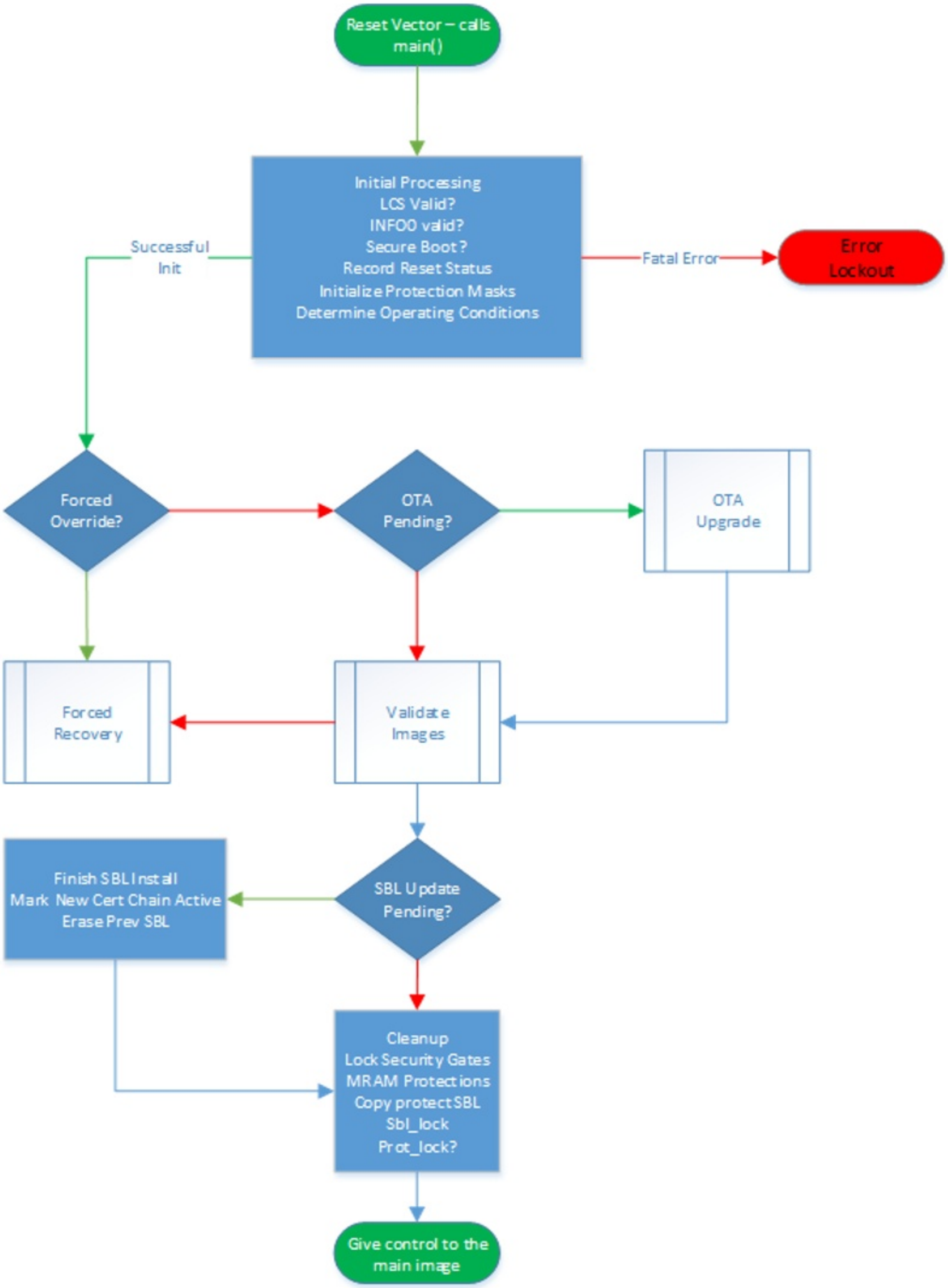
- Authentication of SBL using the ICV Cert Chain
- ICV/OEM Provisioning support
- Life Cycle Management
- Secure Debug Certificate Processing
- Process SBL OTA Updates

3.1.2 Secure Boot Loader (SBL) (Upgradeable)

- Authentication of Ambiq and Customer images
- Secure updates using OTA images
- Value Added Services
  - Support for Non-secure and Secure mode of operation
  - Enforcement of INFOC-OTP security policies
  - Wired Updates over UART or SPI
  - Trim Updates
- SBL Updates (passed to and processed by the SBR)

The SBL can be in-field updated using secure OTA. The overall SBL flow is shown in Figure 3-2 on page 15.

Figure 3-2: Secure Boot Loader (SBL) Flow



3.1.3 Customer Secure Bootloader

The OEM can extend the SBL by providing a secondary bootload that can add support for additional OEM defined features such as:

- Enhanced/Custom Services
- Proprietary Cryptographic Algorithms
- External Memory Device Drivers
- Custom Interface Drivers or Protocols

SECTION

4

Security Life Cycle States

4.1 Chip Manufacturer (CM) LCS

Ambiq acting with the fabrication, packaging and test house partners is considered the Chip Manufacturer. In CM LCS, the SoC is mostly open and ready for initial provisioning. In CM LCS, Ambiq provisions their secret assets (root keys), the Root of Trust and the ICV security policy.

4.2 Device Manufacturer (DM) LCS

Ambiq's customers are considered Device Manufacturers. DM LCS is the initial state of the Apollo5 devices when received by customers. Ambiq provides tools that allow customers to provision their own secret assets, Root of Trust, and security policy settings. The customer can elect to operate the Apollo5 in non-secure mode and stay in DM LCS perpetually or they can operate in secure mode in Secure (SE) LCS.

4.3 Secure Enabled (SE) LCS

SE LCS is the state when the SoC is fully secured and content is authenticated and updated according to the established customer configured security policy. Debugging is disabled by default. This is the proper state of a fully secure device "in the field."

4.4 Return Merchandise Authorization (RMA) LCS

RMA LCS is activated when a faulty SoC needs to be returned to Ambiq for analysis and debug. RMA LCS is a special state and does not allow the recovery of the SoC and cannot be returned to service. As RMA LCS is entered both Ambiq and the OEM security assets are erased.

SECTION 5 Secure Boot Mode

The Apollo5 SoC defaults as a "Secure" SKU, although it may be released as a "Non-Secure" SKU in the future. For the Secure SKU, there are two security "modes" which can be provisioned by the customer using INFOC-OTP, as part of provisioning:

- Non-Secure Mode
  - Secure boot and some other secure services are not supported.
- Secureboot Mode
  - Secure boot and other secure services are supported.

An unprogrammed chip defaults to Non-secure mode.

The Apollo5 can be configured for Secureboot mode in both DM LCS, and Secure LCS. When configured for secure boot, SBL validates the OEMs boot image using a OEM installed certificate chain, and any tampering or corruption of the image would result in a boot failure.

> **NOTE:** Ambiq installed assets (e.g., SBL), are always verified using the same certificate chain approach, irrespective of whether Secure Mode is enabled or not.

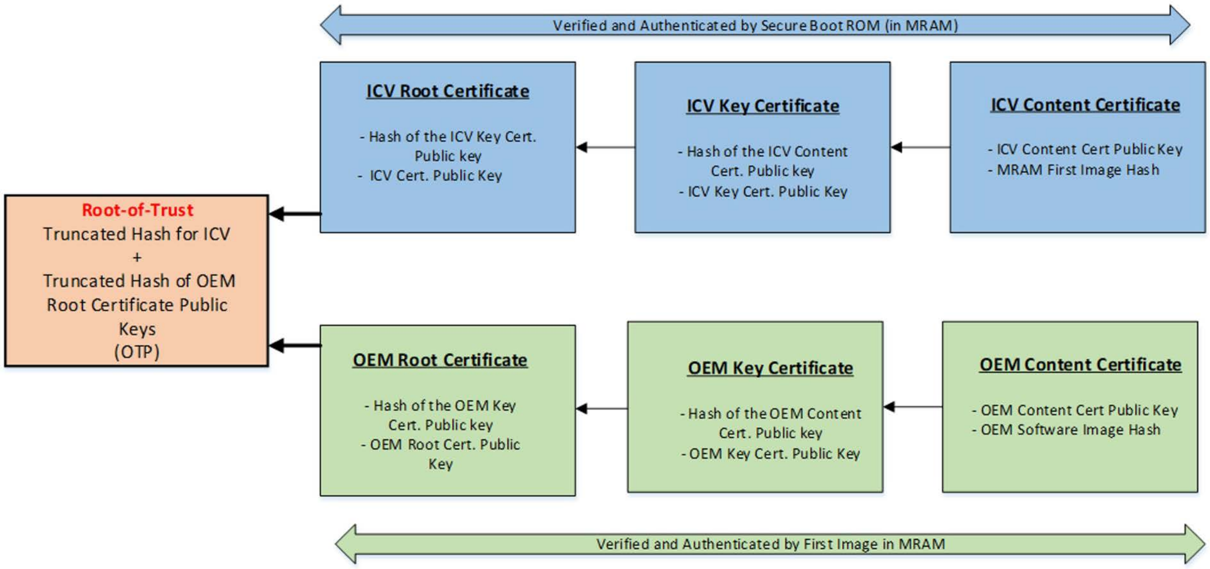
5.1 OEM Certificates and Certificate Chains

The Apollo5 security features support certificate chains for OEM content management as shown in Figure 5-1 on page 19. The Root-of-Trust for these chains is the same truncated public key hash located in INFOC-OTP that’s mentioned above.

There are three certificates in the chain:

- OEM Root Certificate
- OEM Key Certificate
- OEM Content Certificate

Figure 5-1: Supported Certificate Chains for OEM Content Management


Verified and Authenticated by Secure Boot ROM (in MRAM)
ICV Root Certificate
- Hash of the ICV Key Cert. Public key
- ICV Cert. Public Key
ICV Key Certificate
- Hash of the ICV Content Cert. Public key
- ICV Key Cert. Public Key
ICV Content Certificate
- ICV Content Cert Public Key
- MRAM First Image Hash
Root-of-Trust
Truncated Hash for ICV
+
Truncated Hash of OEM Root Certificate Public Keys
(OTP)
OEM Root Certificate
- Hash of the OEM Key Cert. Public key
- OEM Root Cert. Public Key
OEM Key Certificate
- Hash of the OEM Content Cert. Public key
- OEM Key Cert. Public Key
OEM Content Certificate
- OEM Content Cert Public Key
- OEM Software Image Hash
Verified and Authenticated by First Image in MRAM

5.1.1 Verifying the Certificate Chain

Use the following procedure to verify the certificate chain:

1. Retrieve the public key from the certificate and calculate its hash.
   - For the OEM Root Certificate compare the hash to the HBK1.
   - For all the other certificates, compare the hash value to the value calculated and stored from the previous certificate.
2. Verify the RSA signature:
   a. Calculate the hash of the certificate excluding the signature.
   b. Use the public key in the certificate to decrypt the signature, recreating the hash value.
   c. Compare the two hash values.
3. Check the software version information.

> **NOTE:**
> - The software version for the chain must be equal or larger than the software version stored in INFOC-OTP.
> - When in DM LCS, the verification step binding the Root Certificate to the RoT in INFOC-OTP is skipped, as RoT is not programmed yet.

5.2 Certificate Formats

5.2.1 Key Certificates

The following structure is used for OEM Root and Key certificates.

Table 5-1: OEM Root and Key Certificate Structure

Structure	Field	Description
Certificate Header	magicNumber	Magic number to validate the certificate.
	certVersion	Certificate version to validate the certificate.
	certSize	Offset in words to the Certificate signature. And number of software components, if any exist.
	certFlags	Bit field interpreted according to the certificate type.
Certificate Public Key	N	3072-bit public key in big endian format
	Np	160-bit Barrett n' value
Certificate Body	swVer	Software version associated with certificate
	nextPubKeyHash	SHA256 hash result
Certificate Signature	sig	3072-bit RSA PSS signature

5.2.2 Content Certificate

The Content Certificate contains information about one or more images to be verified. The certificate contains an entry for each image, which has the address of the image in NVM, the size of the image, and the Hash corresponding to the image (for authentication). By convention, the content certificate Image[0] always corresponds to the main application image. Thus, upon successful verification, control is passed to this image.

The Content Certificate can also contain optional protection configuration, instructing the SBL to apply either copy or write protection attributes to the images before handing off control to the main image.

Table 5-2: Content Certificate Structure

Structure	Field	Description
Certificate Header	magicNumber	Magic number to validate the certificate.
	certVersion	Certificate version to validate the certificate.
	certSize	Offset in words to the Certificate signature. And number of software components, if any exist
	certFlags	Bit field interpreted according to the certificate type.
Certificate Public Key	N	3072-bit public key in big endian format
	Np	160-bit Barrett n' value

Table 5-2: Content Certificate Structure *(Continued)*

Structure	Field	Description
Content	swVer	Software version associated with certificate
	nonce	Pseudo-random arbitrary number
	imageRec[]	Array of image records
	imageHash	SHA256 hash result
	loadAddr	MRAM address to load image
	imageMaxSize	Max Image size in bytes
	isAesCodeEncUsed	Image is encrypted (Not Supported)
	CopyProtect	Protect the MRAM sectors from Read
	WriteProtect	Protect the MRAM sectors from Write
Certificate Signature	sig	3072-bit RSA PSS signature

SECTION

6

Debug Support

The Apollo5 has hardware support for selective enabling of pre-identified debug features using Arm Cryptocell Debug Control Unit (DCU).

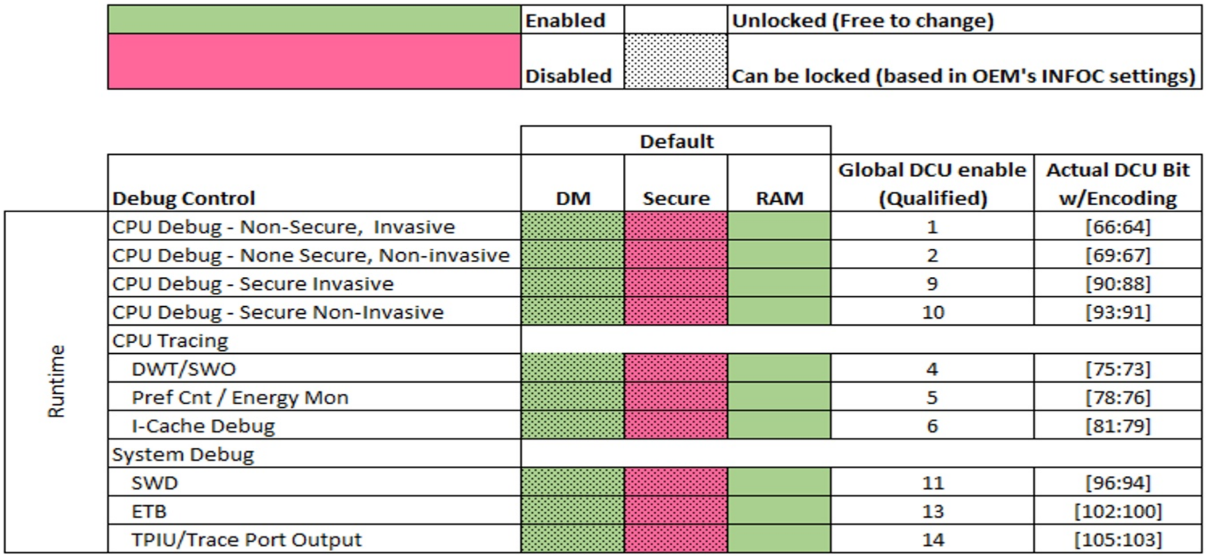
DCU controls are defined as 'qualified' bitmasks – with predefined significance for bits (see Global DCU Enabled in Figure 6-1). At the hardware level, each qualified bit is triple encoded for added security (raw DCU mask).

Hardware raw DCU bits Raw DCU bits (CRYPTO->HOSTDCU*2-3) are triple encoded: b’101 enables a feature, while b’010 disables it.

The following controls are available for the raw DCU masks:

- Enable Bitmask (control individual debug feature accessibility)
  - Fixed Default Values based on LCS
- Lock Bitmask (once locked, the corresponding DCU bit cannot be modified until POI)
  - INFOC-OTP configuration to lock the state of DCU (CRYPTO->LOCKMASTER*2-3)

Figure 6-1: Table for LCS Defaults



🟩	Enabled		Unlocked (Free to change)
🟥	Disabled	▨	Can be locked (based in OEM's INFOC settings)

		Default				
	Debug Control	DM	Secure	RAM	Global DCU enable (Qualified)	Actual DCU Bit w/Encoding
Runtime	CPU Debug - Non-Secure, Invasive	🟩▨	🟥▨	🟩	1	[66:64]
	CPU Debug - None Secure, Non-invasive	🟩▨	🟥▨	🟩	2	[69:67]
	CPU Debug - Secure Invasive	🟩▨	🟥▨	🟩	9	[90:88]
	CPU Debug - Secure Non-Invasive	🟩▨	🟥▨	🟩	10	[93:91]
	CPU Tracing					
	DWT/SWO	🟩▨	🟥▨	🟩	4	[75:73]
	Pref Cnt / Energy Mon	🟩▨	🟥▨	🟩	5	[78:76]
	I-Cache Debug	🟩▨	🟥▨	🟩	6	[81:79]
	System Debug					
	SWD	🟩▨	🟥▨	🟩	11	[96:94]
	ETB	🟩▨	🟥▨	🟩	13	[102:100]
	TPIU/Trace Port Output	🟩▨	🟥▨	🟩	14	[105:103]

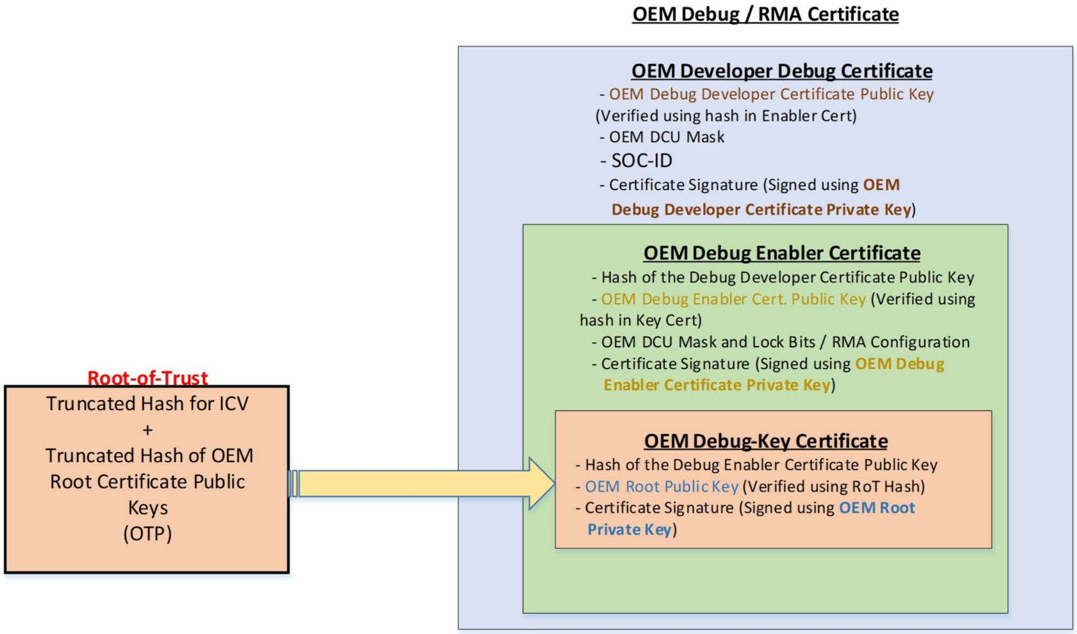
Enabling Debugging of the Apollo5 SoCs While in Secure LCS

In order to override default DCU values (based on LCS) and locking (based on INFOC-OTP configuration), the customer must load a Secure Debug Certificate into NVM.

6.1 Secure Debug Certificates

Secure Debug Certificates are a set of two or three level certificates which contain public keys and authentication values. Figure 6-2 shows the relationship between the certificates.

Figure 6-2: Relationship Between Secure Debug Certificates


OEM Debug / RMA Certificate
OEM Developer Debug Certificate
- OEM Debug Developer Certificate Public Key (Verified using hash in Enabler Cert)
- OEM DCU Mask
- SOC-ID
- Certificate Signature (Signed using OEM Debug Developer Certificate Private Key)
OEM Debug Enabler Certificate
- Hash of the Debug Developer Certificate Public Key
- OEM Debug Enabler Cert. Public Key (Verified using hash in Key Cert)
- OEM DCU Mask and Lock Bits / RMA Configuration
- Certificate Signature (Signed using OEM Debug Enabler Certificate Private Key)
OEM Debug-Key Certificate
- Hash of the Debug Enabler Certificate Public Key
- OEM Root Public Key (Verified using RoT Hash)
- Certificate Signature (Signed using OEM Root Private Key)
Root-of-Trust
Truncated Hash for ICV
+
Truncated Hash of OEM Root Certificate Public Keys
(OTP)

For the two-level SD certificate scheme, the chain order is:

- enabler debug certificate (signed using OEM Root Key) → developer debug certificate.

For the three-level SD certificate chain order is

- key certificate (signed using OEM Root Key) → enabler debug certificate → developer debug certificate.

The Key certificate is protected by the software certificate version (and hence can be revoked).

Ambiq recommends use of three level certificate chain as depicted here.

The function of each certificate in a three level Certificate is as follows:

- OEM Debug Key Certificate
  - Signed by OEM's Root Key (Hash Provisioned in INFOC-OTP)

  - Contains the Hash of Public Key used to verify the Enabler Debug Cert
- OEM Debug Enabler Certificate
  - Signed by Enabled Debug Cert Key (Hash in Key Cert)
  - Contains the Hash of Public Key used to verify the Developer Debug Cert
  - Defines the subset of DCU bits open for editing by Developer
  - Overrides the DCU lock mask (supersedes the INFOC-OTP configuration)
  - Applicable only to the LCS for which it is created
  - Also used for transition to RMA LCS (see next section)
- OEM Developer Debug Certificate
  - Signed by Developer Debug Cert Key
  - Defines the actual DCU overrides
  - Specific to a chip (SOCID) when in Secure LCS

Refer to the *Apollo5 Family OEM Provisioning and Tools User's Guide A-SOCAP5-UGGA03EN* for details on how to generate Secure Debug or RMA Certificates.

6.2 Secure Debug Certificate Handling

As mentioned before, the customer must provide a mechanism to download the Secure Debug Certificate (SD Cert) to a production device as deployed in the field. If enabled, Ambiq's Secure Bootloader provides a means to download an SD Cert using one of the wired interfaces, but the OEM Security Policies are enforced for the download, and can be set to disable this capability.

The location for the SD Certs is specified in the INFO0 configuration. A subsequent Reset after installing the SD Cert will make the Debug overrides effective. Once installed, the SD Certs remain effective, until they are removed. The Debug overrides are in place till the next POI (even if the SD Certs is removed before that).

SD Certs are chip specific, when in Secure LCS creating a SD Cert requires the 256b SOC-ID which is accessible to read using a HAL API function (and is also output via the SBL's SWO output at boot time).

SECTION

7

Run-Time Security Services

AmbiqSuite relies on the Arm CryptoCell engine to provide hardware acceleration for various security services to the main application.

Access to these services is provided through Run-time Software Library – maintained as part of open source effort at TrustZone CryptoCell-312 runtime library from Arm, which can be ported to respective OEM infrastructure. It supports Arm MBed™ TLS APIs to access the underlying hardware acceleration.

- Supported Algorithms: This is a non-exhaustive list of algorithms supported. (Refer to Arm® TrustZone® CryptoCell-312 documentation for reference)
    - Symmetric Key
        - AES 128/192/25
        - DES/TDES 64/128/192
        - AES MAC 128/192/256
        - AES-CCM 128/192/256
    - Asymmetric Key
        - RSA PKCS#1 2048/3072
        - ECC
        - DH
    - Hash / HMAC
        - SHA1, SHA2 (SHA224, SHA256, SHA384, SHA512)
        - MD5
- TRNG (True Random Number Generator)

NOTE: Weak cryptographic algorithms including DES, TDES, SHA1, and MD5 are supported for legacy purposes only. Ambiq recommends against their use.

SECTION

8

Factory Default SoC Configuration

Ambiq ships the Apollo5 SoC in a factory default configuration as follows:

- Device Manufacturing (DM) LCS
- The customer's OTP space is not provisioned and set to 0's
- Device boots in non-secure mode
- Main image assumed at default (0x410000) address
- Debugger is available after the Ambiq bootloader exits
- Debugger is not allowed during Ambiq bootloader execution
- Wired Host connection is enabled, with settings:
  - UART0 - GPIO 30 for TX, GPIO 55 for RX
  - No CTS/RTS
  - Baudrate of 115200
  - Timeout after 500 msec
- INFO0 can be programmed with tools provided in the AmbiqSuite SDK to generate a INFO0.bin (using create_info0.py) and program it into INFO0 using a Jlink commander script (**jlink-prog-info0.txt** and **jlink-prog-info0_otp.txt**)

Default Factory parts boot in non-secure mode without any further configuration. The main image (or secondary bootloader) can be programmed and run from default (0x410000) address in non-volatile memory. The device can be used for development and non-secure testing without any further provisioning, INFO0 can be programmed – but it is not necessary for initial development.

The INFO0 and INFOC-OTP programming overrides many of the default settings to allow the customization of security policy and assets.

SECTION

9

SecureSPOT Configuration

There are two memory areas that are open to the customer for configuration of the SecureSPOT features: INFOC which is One Time Programmable (OTP) and INFO0, that can be selected to be either reprogrammable (MRAM based) or OTP.

9.1 INFOC-OTP Memory

INFOC is one-time programmable (OTP) memory. Transitioning INFOC-OTP area bits from 0 to 1 is a one way operation. Bits can be programmed from 0 to 1 while in DM LCS, but can't be programmed back to 0. INFOC will permanently retain its programming configuration for the life of the part.

INFOC is typically configured in a batch process as part of provisioning to Secure LCS, but the words in INFOC can also be partially or individually, programmed anytime while still in DM LCS. All INFOC area fields that get programmed as part of provisioning, are writable in DM LCS and thus can be provisioned programmatically (through a HAL API function) if desired[1].

Note that the last item provisioned is typically the RoT as that results in the transition from DM-LCS to Secure-LCS (SE-LCS) and in SE-LCS most of the INFOC-OTP fields are no longer writable.

Note also that when a device is changed to RMA-LCS some of INFOC-OTP fields will be erased (by setting all bits to '1'). This is done as a security measure to protect OEM assets and protect against tampering and other types of intrusions.

The most up to date documentation of the OTP fields is provided along with the AmbiqSuite SDK here:

/docs/registers/Apollo510/pages/am_mcu_Apollo510_otpinfoc.html

[1] As long as the partial provisioning configuration is subset of full provisioning data, (any bits set to 1 can not be changed back). A partially provisioned part can accept the additional provisioning steps to complete the Secure LCS transition.

All the register names in INFOC-OTP start with **OTP_INFOC_** which will be dropped from the names in the remainer of this section (e.g., Root of Trust is **OTP_INFOC_ROT0**, will be referred to as **ROT0**).

In general, OTP contains the following:

- Root of Trust (RoT)
- Hardware Keys
  - OEM Symmetric Key used to authenticate the provisioning tools (KCP)
  - OEM Symmetric Key used for encryption/description (KCE)
- OEM Security Policy Selections (**SECURITY** and **SEC_POL**)
- Debug Control Unit (DCU) Overrides (**DCU_DISABLEOVERRIDE**)
- Wired Update Configuration (**BOOT_OVERRIDE** and **WIRED_CONFIG**)
- OEM Key Assets

Further documentation of the meaning of these fields is contained in *Section 14 Appendix on page 49*.

9.2 OEM Keys/Secure Assets

This section proves details of specific fields in INFOC. As in the previous section, for brevity, the register names in this section will drop the OTP_INFOC prefix.

9.2.1 Asymmetric Root of Trust

The Apollo5 SecureSPOT requires the OEM to generate a key pair which in turn is used to generate the OEM portion (128-bits) of the SoC Root-of-Trust. This key pair is only known to the customer, not Ambiq and ensures that the customer has complete and final control over their own SoC security. Ambiq provides tools which accept the public key from the key pair and generates a truncated hash which is stored in INFOC-OPT.

Table 9-1: Asymmetric Root of Trust Field Descriptions

Field Name	Description
ROT4-7	This field is the 128 bit OEM component of Root of Trust Words (HBK1)

9.2.2 Symmetric Keys

The Apollo5 SecureSPOT requires two symmetric keys to be provisioned into INFOC-OTP. The first key KCP is a 128-bit root key designed to authenticate the OEM provisioning tool to the Apollo5 SoC. The second symmetric key KCE is another 128-bit root key designed to be used for OEM firmware image decryption. These keys cannot be read after the OEM provisions the device in DM LCS. They can be loaded into AES hardware engine directly for symmetric key crypto operations.

Table 9-2: Symmetric Keys Field Descriptions

Field Name	Description
KCP0-3	128 bit AES Key (referred to as Provisioning Key – KCP)
KCE0-3	128 bit AES Key (referred to as Code Encryption Key – KCE)

9.2.3 Custom Key Storage

In addition to the required secure assets, the Apollo5 provides additional key storage area (255Bytes) that can be used to store any OEM specific secrets. The areas are divided into four quadrants. The access policy for each quadrant is individually controlled by the customer. The customer can allow access to each quadrant by Ambiq's Secure Bootloader (SBL). The customer can also allow access through a designated Keybank Access Key, which is programmed during provisioning. Quadrants 0 and 1 are reserved for static keys programmed as part of DM provisioning. Quadrants 2 and 3 can either be used for static keys or may be optionally used for run time generated, one-time programmable (OTP) keys.

Table 9-3: Custom Key Storage Field Descriptions

Field Name	Description
KEYBANK_CUST_QUADRANT0_KEY0-15	Customer Keybank Quadrant 0
KEYBANK_CUST_QUADRANT1_KEY0-15	Customer Keybank Quadrant 1
KEYBANK_CUST_QUADRANT2_KEY0-15	Customer Keybank Quadrant 2
KEYBANK_CUST_QUADRANT3_KEY0-15	Customer Keybank Quadrant 3

See *Section 14 Appendix on page 49* for more information about the OEM Key Storage area.

9.3 OEM Security Policy

This section describes the various settings that customers can use to establish a customized security policy that meets their product's needs. Some of the settings are in INFO0 space while others are in INFOC-OTP. Some of the fields are triple bit encoded to make electrical attacks more difficult. See *Section 14 Appendix on page 49* for more information about these fields.

9.3.1 Configuration of Secure Boot

Table 9-4: Secure Boot Field Descriptions

Field Name	Description
SECURITY : CUST_SECBOOT	This field determines if Secure Boot mode is enabled. Note that this is distinct from the SE LCS state, which is a lifecycle state. Devices in DM as well as SE LCS can be configured for nonsecure boot, or secureboot based on this configuration. Secureboot enabled devices go through a certificate chain based verification for the image by Ambiq SBL using the CERTCHAINPTR programmed into INFO0.
SECURITY : CUST_SECBOOTONRST	This field should only be used if Secure Boot (CUST_SECBOOT) is enabled. This field determines if the Secure Bootloader will execute the entire secure boot flow on every reset event. Selection of this option may cause additional latency on resets.

9.3.2 Secondary Bootloader Configuration

Table 9-5: Secondary Bootloader Field Description

Field Name	Description
SECURITY : PLONEXIT	If configured to not lock it (value 0), the Secure Bootloader (SBL) keeps PROTLOCK open for a customer secondary bootloader. This allows access to customer keybank area, and in addition, PROTLOCK is used to deny/allow writing to the "Flash" (NVM) protection registers. SECURITY:PLONEXIT should be set, if no customer secondary bootloader present. This ensures that the main images do not have the ability to change NVM protection. Moreover, it also locks the INFOC-OTP keybank from general access once the SBL exits. When a secondary bootloader is present then, the secondary Bootloader should assert the PROTLOCK upon exit: MCUCTRL->BOOTLOADER_b.PROTLOCK (write 1 to clear) It is recommended that this be done by Secondary Bootloader on exit, but may be optionally kept open if the entire customer application is trusted.

9.3.3 INFO Space Protections

Table 9-6: INFO Space Protections Field Description

Field Name	Description
SECURITY : DIS_CUST_INFO_PROG	This field allows the customer to selectively disable write access to any or all of the four quadrants of INFO0-MRAM space. INFO0-OPT is protected by the same bit that protects the first quadrant of INFO0-MRAM.

9.3.4 Update Configuration

Table 9-7: Update Configuration Field Description

Field Name	Description
SEC_POL : AUTH_ENF_ECC	This field forces authentication of customer updates even if there is an Embedded Content Certificate (ECC) included with the update.
SEC_POL : ENC_ENFORCE	This field determines whether decryption is required for all image updates.
SEC_POL : AUTH_ENFORCE	This field determines whether authentication is required for all image updates.

9.3.5 SWO Output Pin Number

Table 9-8: SWO Pin Assignment

Field Name	Description
SEC_POL : SBL_SWO_PIN	Pin number for SWO output to use. Default is pin 28, setting this to 0xFF turns SWO output off.

9.3.6 Wired Interface Configuration

Table 9-9: Wired Interface Configuration Field Description

Field Name	Description
BOOT_OVERRIDE : GPIO	This field sets the GPIO pin to allow an external device to override the secure boot flow and force processing of a wired update.
BOOT_OVERRIDE : POL	This field sets the polarity of the GPIO override pin as active high or low.
BOOT_OVERRIDE : ENABLE	This field enables the GPIO override feature. By default it is disabled.
WIRED_CONFIG : UART WIRED_CONFIG : SPI	These fields enable UART, or SPI interfaces for wired update operation. The customer may enable one or more of these interfaces as needed, but typically will choose only one.
WIRED_CONFIG : SLAVEINTPIN	This field selects the GPIO to be used by an external device to interrupt the IOS if SPI mode is enabled.
WIRED_CONFIG : UARTMODULE	This field selects the UART instance/module (0-3) to be used for the wired interface if UART mode is enabled. Specific UART configurations are controlled through INFO0.

9.3.7 NVM Access Protection

The maximum NVM space for Apollo510 is 4MB. For each of the protection fields in INFOC-OTP there are eight (8) 32-bit words or 256 bits total assigned to protect regions/sections of NVM. This means that the customer may protect memory blocks of 16K each. Bit 0 is the first 16K block (offset 0x00000000 – 0x00003FFF) and Bit 256 is the last 16K block (offset 0x003FC000 to 0x003FFFFF).

Table 9-10: NVM Access Protection Field Description

Field Name	Description
CUST_WPROT CUST_RPROT	These fields are controlled by the customer during provisioning. They provide permanent write and read protection. WPROT provide write protection to the specific NVM 16K blocks (e.g., "Chunks"). RPROT provide "copy" or read protection. See *Section 14 Appendix on page 49* for more information. Note these protections should not used when MRAM recovery is deployed. The areas protected cannot be restored by the MRAM recovery process as it requires both read and write access to any areas to be recovered. Also. protections for the lower 64K of the NVM memory must never be enabled (e.g., do not set lower 4 bits (bit0-bit3) of CUST_WPROT0 or CUST_RPROT0) as it would interfere with the functioning of the pre-installed Secure Bootloader (SBL).
SBL_WPROT SBL_RPROT	Similar to CUST_*PROT. However, while protected from general access, write/access to these NVM blocks is possible through SBL via a secure update and during the MRAM Recovery process.

9.3.8 Symmetric Keybank Protections

Table 9-11: Symmetric Keybank Protection Field Description

Field Name	Description
CUSTOTP_PROGLOCK CUSTOTP_RDLOCK	These values determine the read/program access to the four quadrants of OEM/customer key banks in INFOC-OTP. Quadrants #0 and #1 cannot be programmed after exiting DM-LCS until moving to RMA-LCS (and hence are used for static keys), Quadrants #2 and #3 can be programmed from the main application or locked down. These can then be optionally used to store run-time generated keys. ■ RDLOCK[3:0] locks down read access to the OEM/customer key banks when running boot operations (e.g., SBR, SBL, and Secondary Bootloader). ■ RDLOCK[7:4] locks down read access to the OEM/customer READ key banks after boot. ■ If not locked, the key banks could be accessed by an authorized software, as long as it has access to a 128-bit unlock key, as programmed in CUSTOTP_READ_KEY[1] ■ PROGLOCK[1:0] lock down access to programming OEM/customer key banks (Quadrant #2 and #3) when running the boot operations. ■ PROGLOCK[3:2] lock down access to programming OEM/customer key banks (Quadrant #2 and #3) after boot. ■ If not locked, these key banks can be programmed by an authorized software, as long as it has access to a 128-bit program key, as programmed in CUSTOTP_PROG_KEY[1]

[1] Implies multiple registers (CUSTOTP_PROG_KEY0 to CUSTOTP_PROG_KEY3).

Table 9-12: LCS Conditions and Results

LCS	Condition	Result
CM, DM, SE	(CUST_RDLOCK[0] == 0) AND ((SBRLOCK == 1) OR (SBLLOCK == 1) OR (PROTLOCK == 1))	Quadrant #0 Key Bank can be read during boot.
CM, DM, SE	(CUST_RDLOCK[4] == 0) AND (CUSTOTP_READ_KEY0 != 0)	Quadrant #0 Key Bank can be read after boot using the master read key.
DM, RMA	N/A	Quadrant #0 Key Bank can be programmed.
CM, DM, SE	(CUST_RDLOCK[1] == 0) AND ((SBRLOCK == 1) OR (SBLLOCK == 1) OR (PROTLOCK == 1))	Quadrant #1 Key Bank can be read during boot.
CM, DM, SE	(CUST_RDLOCK[5] == 0) AND (CUSTOTP_READ_KEY0 != 0)	Quadrant #1 Key Bank can be read after boot using the master read key.
DM, RMA	N/A	Quadrant #1 Key Bank can be programmed
CM, DM, SE	(CUST_RDLOCK[2] == 0) AND ((SBRLOCK == 1) OR (SBLLOCK == 1) OR (PROTLOCK == 1))	Quadrant #2 Key Bank can be read during boot.
CM, DM, SE	(CUST_RDLOCK[6] == 0) AND (CUSTOTP_READ_KEY0 != 0)	Quadrant #2 Key Bank can be read after boot using the master read key.
DM, RMA	(CUST_PROGLOCK[0] == 0) AND ((SBRLOCK == 1) OR (SBLLOCK == 1) OR (PROTLOCK == 1))	Quadrant #2 Key Bank can be programmed during boot.
DM, RMA	(CUST_PROGLOCK[2] == 0) AND (CUSTOTP_PROG_KEY !=0)	Quadrant #2 Key Bank can be programmed after boot using the master programming key.
CM, DM, SE	(CUST_RDLOCK[3] == 0) AND ((SBRLOCK == 1) OR (SBLLOCK == 1) OR (PROTLOCK == 1))	Quadrant #3 Key Bank can be read during boot.
CM, DM, SE	(CUST_RDLOCK[7] == 0) AND (CUSTOTP_READ_KEY0 != 0)	Quadrant #3 Key Bank can be read after boot using the master read key.
DM, RMA	(CUST_PROGLOCK[1] == 0) AND ((SBRLOCK == 1) OR (SBLLOCK == 1) OR (PROTLOCK == 1))	Quadrant #3 Key Bank can be programmed during boot.
DM, RMA	(CUST_PROGLOCK[3] == 0) AND (CUSTOTP_PROG_KEY !=0)	Quadrant #3 Key Bank can be programmed after boot using the master programming key.

NOTE: The expression ((SBRLOCK == 1) OR (SBLLOCK == 1) OR (PROTLOCK == 1) implies that the code is running in the Bootloader and not yet reached the main application.

9.3.9 Debug Control

The Apollo5 SoC supports fine granularity debug control through a hardware Debug Control Unit (DCU). There are distinct debug features which are controlled by individual DCU bits. As can be seen in *Section 14 Appendix on page 49*, debug control defaults differ based on the LCS.

Hardware raw DCU bits are triple encoded: b'101 enables a feature, while b'010 disables it.

Table 9-13: Debug Control Field Description

Field Name	Description
LOCKMASK[0-3]	The DCU values can be locked and prevented from further modifications by configuring and specifying the triple bits corresponding to the feature to be locked.
DCU_DISABLEOVERRIDE	This field provides the ability to disable a feature upon SBL exit, even if the DCU settings allow for the feature. Setting a bit in the 21-bit field forces the desired debug feature to be disabled. Unlike the DCU locking, this override can be reversed by programming MCUCTRL->DEBUGGER register.

9.4 OEM Infospace (INFO0) Configuration

INFO0 provides storage which is separate from the main MRAM. Some of the fields in INFO0 are predefined and allow the OEM to override/configure the default settings provisioned in INFOC. It also allows customers to override the defaults for certain attributes of the boot flow.

The Apollo5 device has two "INFO0" spaces, one that is based on MRAM (rewritable) and one that is OTP (one time programmable). Only one of which is active at any one time, providing INFO0 configuration data to the device, but either can be read/written at any time. Which one is active is determined by the selector register in INFOC (**SHDW_TRIM_INFO0_SEL** at 0x400C23FC). See the register documentation for this register and its use.

INFO-OTP is 256 bytes in size and INFO0-MRAM is 2048 bytes (2K) in size. The first ~160 bytes in each has Ambiq defined functionality (that are the same for the INFO0-MRAM or INFO0-OPT), with the remaining space in each available for the customer's use. Ambiq's scripts and tools create INFO0 binary images that are 256 bytes in size, and the same image is used to program either (or both) INFO0-MRAM or INFO0-OTP. The remaining space in INFO0-MRAM is available for the customer's use.

INFO0-MRAM is split into four quadrants, which can be individually write protected through INFOC security settings (in the SECURITY word). INFO0-OTP is protected from being written by the same bit that protects the first quadrant of INFO0-MRAM.

9.4.1 Signature

Table 9-14: Signature Field Description

Field Name	Description
INFO0_SIGNATURE	This field uses four (4) 32-bit words to form a 128-bit signature. All of INFO0 is left uninitialized (set to 0x0) during Ambiq's manufacturing. INFO0 provisioning will set the signature to indicate a valid customer INFO0 configuration. The same signature is used for both INFO0-MRAM and INFO0-OTP.

9.4.2 Customer Trims

Table 9-15: Customer Trims Field Description

Field Name	Description
INFO0_CUSTOMER_TRIM	This field must be set to '0' due to an Errata (ERR079).

9.4.3 Wired UART Config

Table 9-16: Wired UART Config Field Description

Field Name	Description
INFO0_SECURITY_WIRED_IFC_CFG0	This word contains the configuration used to configure the UART module specified in the OTP_INFOC_WIRED_CONFIG field.
INFO0_SECURITY_WIRED_IFC_CFG1	This word contains the 8-bit GPIO pin numbers for up to four pins that may be used for UART RX, TX, CTS, RTS pins.
INFO0_SECURITY_WIRED_IFC_CFG2 INFO0_SECURITY_WIRED_IFC_CFG3 INFO0_SECURITY_WIRED_IFC_CFG4 INFO0_SECURITY_WIRED_IFC_CFG5	These words contain the GPIO Pin Config field used for each of the pins defined in WIRED_IFC_CFG1 field above in the same order specified.
INFO0_WIRED_TIMEOUT	Because the UART interface is asynchronous, the Secure Bootloader waits for a connection from the external host for this amount of time (in ms) before it exits the wired update process.

9.5 Security Version

Table 9-17: Security Version Field Description

Field Name	Description
INFO0_SECURITY_VERSION	This field contains the Version ID for INFO0 and is just for tracking purpose.

9.5.1 Memory Reservation

Table 9-18: Memory Reservation Field Description

Field Name	Description
INFO0_SECURITY_SRAM_RESV	This word determines the amount of CPU SRAM (DTCM) to reserved for application scratch space. This reserves the specified memory at the top end of CPU's SRAM memory address range (e.g., 0x2007FFFF). This memory will not be disturbed by the Secure Boot Loader during any of the boot or update operations. The starting point of the reserved memory is 0x2008000 – SRAM_RESV. Note: This space will not be retained across a MRAM recovery operation, POI, or power cycle operations, and would need to be reloaded.

9.5.2 Enable RMA Override

Table 9-19: RMA Override Field Description

Field Name	Description
INFO0_SECURITY_RMAOVERRIDE	This field determines if Ambiq has permission to download an Ambiq RMA certificate after the customer has performed their part of the RMA process. Refer to *Section 12 Transition to RMA LCS on page 45* for RMA transition process.

9.5.3 Secure Debug Certificate Locations

Table 9-20: Secure Debug Certificate Locations Field Description

Field Name	Description
INFO0_SBR_SDCERT_ADDR	This word is the location of the of the Secure Debug Certificate in memory. The certificate may be located in user accessible MRAM, DTCM, ITCM or SRAM. Note: If the SD Cert is placed in RAM, it will be lost whenever a POI or power cycle reset occurs.

9.5.4 Main Application Location (Non-Secure Boot)

Table 9-21: Main Application Location (Non-Secure Boot) Field Description

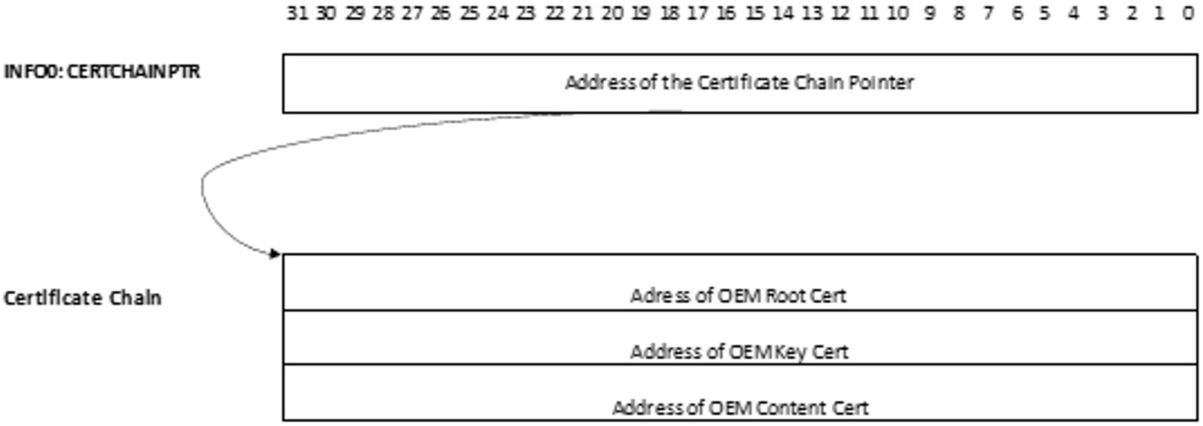
Field Name	Description
INFO0_MAINPTR	This word is the location of the main application image when non-secure boot flow is operational.

9.5.5 Certificate Chain Location (Secure Boot)

Table 9-22: Certificate Chain Location (Secure Boot) Field Description

Field Name	Description
INFO0_CERCHAINPTR	The Apollo5 SoC uses a "chain" of certificates to authenticate images/content during the secure boot flow. This word is the address of the Certificate Chain pointers, which are 3 contiguous words in MRAM memory. These point to the OEM Root, OEM Key, and OEM Content certificates as shown in Figure 9-1.

Figure 9-1: OEM Certificate Chain Pointers



9.5.6 MRAM Recovery Controls (in INFO0)

The reminder of the Ambiq defined locations in INFO0 are for configuring the MRAM Recovery and its options, (15 words). The details of the MRAM recovery feature and it's configuration is covered in detail in the *Apollo5 Family MRAM Recovery User's Guide A-SOCAP5-UGGA01EN*, refer to that document for information on enabling MRAM recovery. When not being used, these locations can be left as their default 0x00000000 values.

SECTION

10

Configuring Secure Boot Mode

The following steps outline the process of configuring the Apollo5 SoC into Secure Boot Mode:

1. Generate the Certificates using the tools provided.
2. Program the Certificate Chain Pointer in INFO0.
3. Program the NVM at the INFO0_CERCHAINPTR address with the addresses to where the three certificates will be loaded.
4. Program all three certificates in NVM at the designated locations in memory.
5. Program the image(s) corresponding to the Content Certificate Image[] list at the designated locations in memory.
6. Configure OTP_INFOC_SECURITY : CUST_SECBOOT to 0x2.
7. Configure OTP_INFOC_SECURITY : CUST_SECBOOTONRST to 0x2 (enable) or 0x5 (disable)
8. Reset the SoC.

SECTION

11 Transition to Secure LCS

Secure LCS is a very restrictive security state in the Apollo5 SoC. The transition to this state locks down resources and restricts many of the normal development pathways (e.g., the debugger). When in Secure LCS, the default DCU settings disable debug access, so a customer cannot talk to the device natively through debugger (Jlink etc). If a subset of debug functionality is desired the customer must install a Secure Debug Certificate with a corresponding configuration prior to the transition to Secure LCS. In the Secure Boot flow, most of the faults and errors detected by SecureSPOT functionality will result in Apollo5 SoC being “locked” by design.

Ambiq recommends that the customer always keep a path open to enable ‘downloading’ by either enabling wired update in the Secure Bootloader or by providing a custom implementation in their main application to update the images in the device. This would be the path to load a Secure Debug Certificate to open up debugging on a device after it has been provisioned to Secure LCS. In addition, selective debug functions could also be enabled in the main application, using the DCU HAL functions, but this will only work if the corresponding DCU flags are not already locked through the INFOC-OTP configuration (as specified by the customers provisioned security options).

In general, it is expected that the customers will do one step provisioning on their production line to set up all the INFOC-OTP fields, be it keys, security options or the RoT itself.

NOTE: The provisioning of RoT triggers the LCS advancement to Secure-LCS upon the next reset.

Subsections below describe the provisioning process in detail. Alternate means using the HAL to program the INFOC-OTP fields is possible but requires a full understanding of the INFOC-OTP structure and details of individual fields that are to be programmed.

The Ambiq supplied OEM provisioning tool consist of two parts::

- Device Provisioning Tool (opt_image_pkg.bin)
  - Used to Provision OEM's encrypted assets securely
  - Provided by Ambiq (Signed with Ambiq's Private key
- Using the PC Tool Chain Example Scripts the OEM:
  - Generates the OEM assets (encrypted or plain unencrypted)
  - Packages the assets into single data blob, used with the opt_image_pkg.bin

11.1 Encrypted vs Plain OEM Assets

The Apollo5 provides two different flows for generating the OEM's Security Assets needed to move the device from DM-LCS to Secure-LCS. The first creates a plain (unencrypted) data blob, and the alternate encrypts the OEM assets using the ICV Key Request and Response from Ambiq that enables the encryption of the Security assets so that the OEMs keys and ROT cannot be compromised if the provisioning tools are being handled by untrusted individuals. For example, when the products are being manufactured and provisioned by a third party in a factory where the provisioning assets may be handled by non-employees.

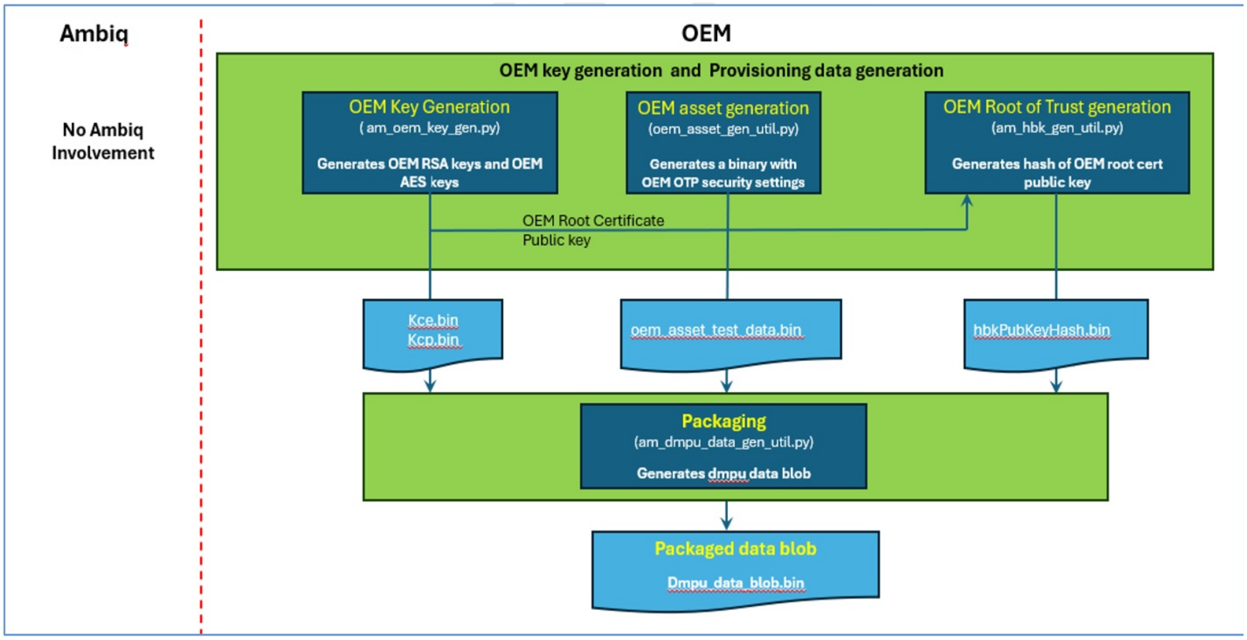
Ambiq provides the tools for each of these steps in the /**tools/Apollo510_scripts/ oem_tools_pkg** directory

11.2 Generating the Provisioning Data Blob

Figure 11-1 is a high-level view of the generation of the OEM Provisioning Blob while in DM LCS state. Ambiq provides the tools for each of these steps in the / **tools/apollo510_scripts/oem_tools_pkg** directory.

Refer to *Apollo5 Family Provisioning Tools User's Guide A-SOCAP5-UGGA03EN* for details on how to use various tools provided with AmbiqSuite SDK to generate various provisioning assets.

Figure 11-1: OEM Plain (Unencrypted) Provisioning Data Blob Generation


Ambiq
No Ambiq Involvement
OEM
OEM key generation and Provisioning data generation
OEM Key Generation
(am_oem_key_gen.py)
Generates OEM RSA keys and OEM AES keys
OEM asset generation
(oem_asset_gen_util.py)
Generates a binary with OEM OTP security settings
OEM Root of Trust generation
(am_hbk_gen_util.py)
Generates hash of OEM root cert public key
OEM Root Certificate Public key
Kce.bin
Kcp.bin
oem_asset_test_data.bin
hbkPubKeyHash.bin
Packaging
(am_dmpu_data_gen_util.py)
Generates dmpu data blob
Packaged data blob
Dmpu_data_blob.bin

Use the following procedure to generate the provisioning blob:

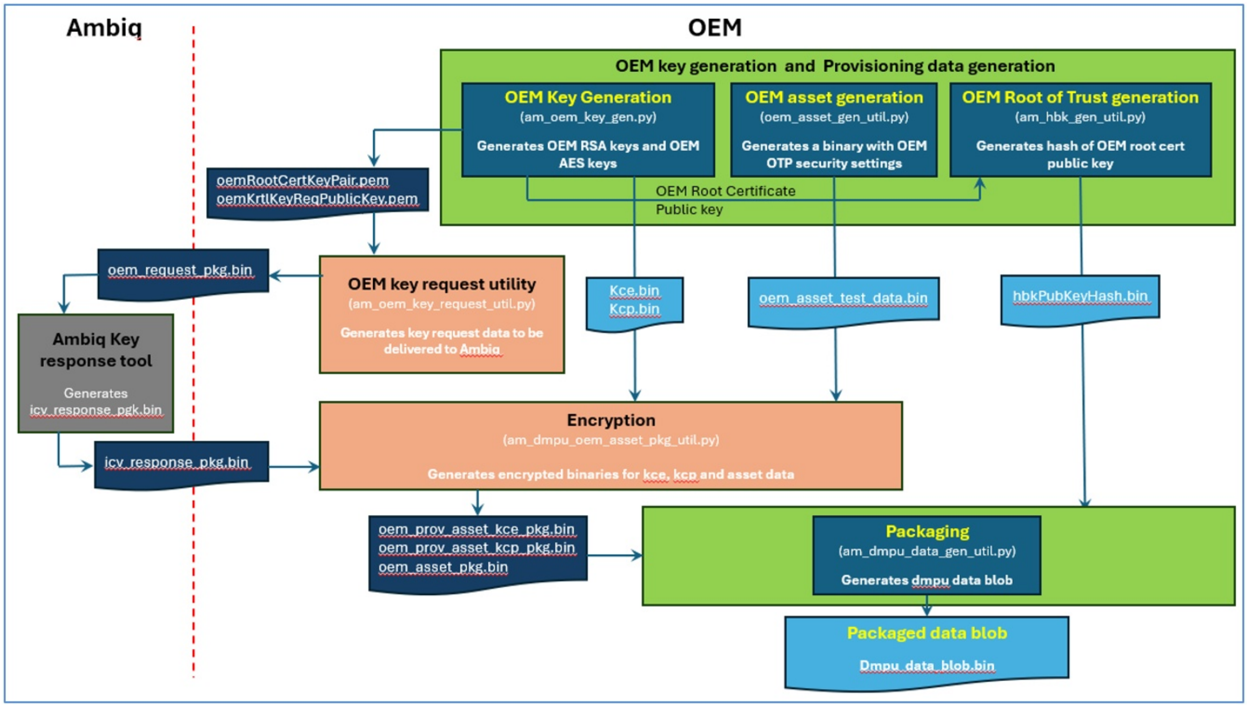
1. Generate Asymmetric and Symmetric Keys
   am_oem_key_gen_util/am_oem_key_gen_util.py
2. Generating RoT (Needed to program in OTP)
   cert_utils/am_hbk_gen/am_hbk_gen_util.py

   Note: Step 1 and 2 can be replaced by the OEM's own established key generation process.
3. Asset Generation, which packages the OEM's INFOC-OTP configurations.
   oem_asset_prov_utils/oem_asset_package/oem_asset_gen_util.py
4. Provisioning Data Generation, which packages all the pieces together to generate the final provisioning data blob.
   oem_asset_prov_utils/oem_asset_package/am_dmpu_prov_data_gen_util.py

The output of the last step is the final "OEM Provisioning Data Blob" sometimes referred to as the dmpu data blob.

Figure 11-2 on page 42 is a high-level view of the generation of the Encrypted OEM Provisioning Data Blob. Ambiq provides the tools for each of these steps in the **/tools/Apollo510_scripts/oem_tools_pkg** directory.

Figure 11-2: OEM Encrypted Provisioning Data Blob Generation


Ambiq
OEM
OEM key generation and Provisioning data generation
OEM Key Generation
(am_oem_key_gen.py)
Generates OEM RSA keys and OEM AES keys
OEM asset generation
(oem_asset_gen_util.py)
Generates a binary with OEM OTP security settings
OEM Root of Trust generation
(am_hbk_gen_util.py)
Generates hash of OEM root cert public key
OEM Root Certificate Public key
oemRootCertKeyPair.pem
oemKrtlKeyReqPublicKey.pem
oem_request_pkg.bin
OEM key request utility
(am_oem_key_request_util.py)
Generates key request data to be delivered to Ambiq
Kce.bin
Kcp.bin
oem_asset_test_data.bin
hbkPubKeyHash.bin
Ambiq Key response tool
Generates icv_response_pgk.bin
icv_response_pkg.bin
Encryption
(am_dmpu_oem_asset_pkg_util.py)
Generates encrypted binaries for kce, kcp and asset data
oem_prov_asset_kce_pkg.bin
oem_prov_asset_kcp_pkg.bin
oem_asset_pkg.bin
Packaging
(am_dmpu_data_gen_util.py)
Generates dmpu data blob
Packaged data blob
Dmpu_data_blob.bin

Note that the process is the same for both, with the encrypted process adding two additional steps:

- OEM Key Request generation (sent to Ambiq)
  ➔ Ambiq ICV Key Response (returned from Ambiq)
- Key Asset Encryption of the OEM data

The additional steps are performed between steps 3 and 4 described above for the plain (unencrypted) procedure:

3.1 Make an Encryption Key Request to Ambiq.
This requests a key from Ambiq, that in combination with the supplied Key Request Public Key, allows for the OEM data blob to be encrypted. The Response from Ambiq is used to encrypt the final provisioning asset blob. To create the ICV Key request, use the following script to generate the key request:

oem_asset_prov_utils/oem_key_request/am_oem_key_request_util.py

3.2 Key Asset Encryption which encrypts the OEM's sensitive asset data so that only they can access it. The following script is used to encrypt the OEMs data.

oem_asset_prov_utils/oem_asset_package/am_dmpu_oem_asset_pkg_util.py

The output of the last step (4) as described in the previous section, creates the "Provisioning Data Blob," but this time the embedded OEM keys are encrypted and are safe to be handled by untrusted third parties.

Other than the two additional steps (3.1 and 3.2) the difference when packaging the Provisioning Data Blob in step 4 is that in the keywords provided in the **.cfg** file for the **am_dmpu_oem_asset_pkg_util.py** script indicate the plain or encrypted status.

- oem-data-type
- kce-data-type
- kcp-data-type

Each of these is set to **= 1** for the plain (non-encrypted) case and **= 2** to indicate the encrypted blob. Examples of both can be found in the example **.cfg** files provided in the AmbiqSuite SDK in the directory:

/oem_asset_prov_utils/oem_asset_package/am_config/

- **am_dmpu_data_gen_nonsec.cfg**
- **am_dmpu_data_gen_plain_nonsec.cfg**

11.3 Provisioning the Device

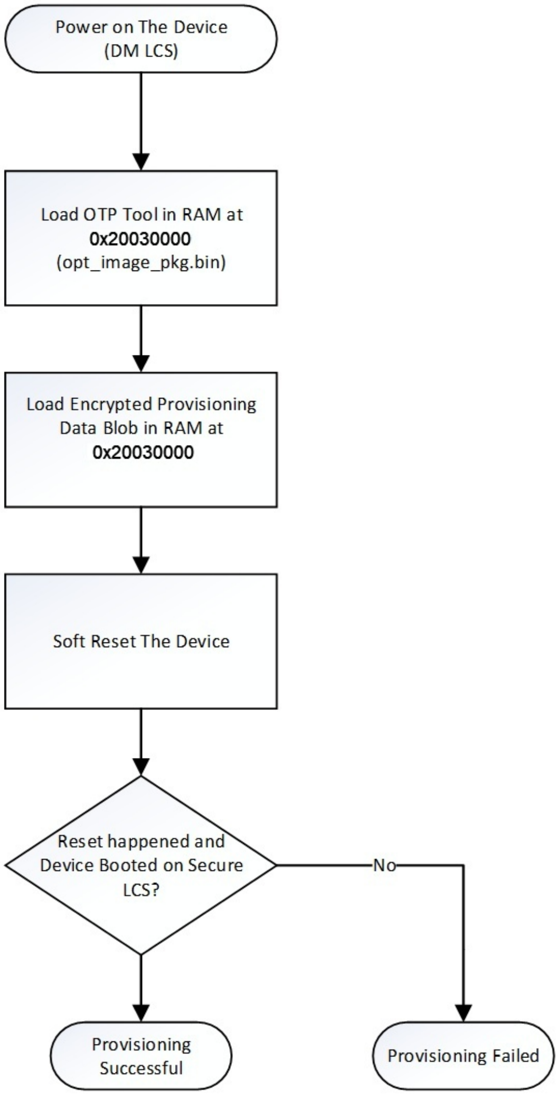
Ambiq provides an OEM Provisioning Tool (OPT) which is an executable binary that can be loaded on to the Apollo5 SoC during customer manufacturing. In addition to this binary, the Encrypted Provisioning Data Blob from the previous section is also loaded into memory.

Use the following procedure to provision the device:

1. Load the OEM Provisioning Tool (OPT) @ 0x20030000
   oem_prov_tool/opt_image_pkg.bin
2. Load the OEM Provision Data Blob @ 0x20037000.
   Note: This is the output of Step 4 from *Section 11.2 Generating the Provisioning Data Blob on page 40*.
3. Perform a POR Reset that will initiate the provisioning and transition the device to Secure LCS.

Figure 11-3 on page 44 shows the flowchart for OEM Provisioning Process.

Figure 11-3: OEM Provisioning Blob Generation



SECTION

12

Transition to RMA LCS

In some situations, it may be necessary to transition the Apollo5 SoC into the RMA LCS. The RMA LCS is a terminal State and is only possible to enter this state at the Ambiq Secure Labs which processes the RMA. All Ambiq and OEM secrets are invalidated permanently during the transition to RMA LCS, including:

- KCE, KCP invalidated
- INFOC-OTP Keybank area & access keys invalidated
- NVM Permanent Copy Protected sectors erased

In order to analyze issues with RMA parts, the Ambiq/customer may be required to create special application images.

Once transitioned to RMA LCS the chip cannot be returned to the customer or be recommissioned.

Successful RMA transition is a two-step process. It requires both a customer signed RMA Cert, as well as an Ambiq Signed RMA Cert to be installed to complete a successful transition to RMA LCS.

The transition to RMA LCS uses the same SD Cert infrastructure as previously described.

AmbiqSuite SDK provides reference scripts and configurations to assist customer generating OEM RMA Certs.

After installation and processing of the OEM RMA Cert, the customer has two options to enable Ambiq to complete the RMA transition:

- Enable SWD debug access to Ambiq (Either enabled by main image, or by an SD Cert installed after the OEM's RMA processing), that would allow the installation of the Ambiq RMA Cert.
- The INFO0 field **INFO0_SECURITY_RMAOVERRIDE** when enabled, allows Ambiq to use SBL wired download to load the Ambiq Signed RMA Cert after the device has already gone through the OEM's RMA processing.

Use the following procedure to transition to RMA LCS are:

1. Install and process the OEM RMA Cert
2. Provide the Apollo5 SoC to Ambiq, enabling one of the two options above to allow Ambiq to install its RMA Cert when it arrives at the Ambiq facility.
3. Provide following additional information along with the chip to Ambiq:
    - SOCID (if Secure LCS)
    - Info0 configurations:
        - SD Cert location
        - Main image (nonsecureboot), or Main CertChain Pointer (secureboot) location
    - Cert version counters – Content of registers at 0x400C2088 to 0x400C2098, from INFOC-OTP

SECTION

13

Image Updates

The Apollo5 secureSPOT supports image updates via customer applications (Firmware Over-the-Air) or via the wired update interface. Additionally, reference JLink Commander scripts are provided with the SDK to assist is loading updates in debug/production-line settings when debugger is enabled.

The following updates are supported:

- Firmware
  - Nonsecure
  - Secure
- INFO0 (either MRAM or OTP)
- Certificate Chain (Primarily used to update the version number, for revocation)
- Trim Patches (Ambiq Provided)
- Secure Bootloader (SBL) Updates
- Key revocation (To revoke INFOC-OTP Keybank keys)

Additionally, Wired Download through the SBL wired update interface is also supported—which supports both raw downloads, as well as downloading an OTA image blob, and in turn initiating regular OTA (update) cycle.

Details of the Blob formats and processing are provided in the *Apollo5 Family Secure Update User's Guide A-SOCAP5-UGGA02EN*.

Details on how to use the AmbiqSuite tools to generate and update the assets are provided in the *Apollo5 Family Provisioning Tools User's Guide A-SOCAP5-UGGA03EN*.

13.1 General Update (OTA) Process

In general, the authentication of the update image blob uses a Public Key based on the boot certificate chain) when required by the security policy. In addition, in-place decryption may also be required by the security policy.

Subsequent Processing depends on the type of image:

- Firmware Updates will install the new firmware at designated location in NVM.
- Secure Firmware Updates will verify the certificate chain to ensure once installed it does not fail to boot, then install the image(s,) and also install a new content certificate, if needed
- Info0 Updates will update INFO0 (MRAM and/or OTP) based on provided information
- Trim Patch Updates will update Device trims by executing an embedded trim patching function (provided by Ambiq)
- Cert Chain Update will update the installed OEM certificates used for secureboot verification.
- Key revocation update is used to update the OEM keybank to revoke specified symmetric keys if they are suspected of being compromised.

AmbiqSuite SDK provides reference JLink scripts for general OTA updates in:
tools/apollo510_scripts/jlink-ota.txt

13.2 Secure Bootloader Image Updates

Installation of an updated Secure Bootloader (SBL) uses a similar OTA update process, with Ambiq providing an encrypted and signed OTA binary image containing the updated SBL. Unlike the other OTA images that are processed by the SBL itself, the SBL update is processed by the Secure Bootrom (SBR).

Released SBL update image files are found in:
/tools/apollo510_scripts/sbl_updates/

Each update release will have a subdirectory with the SBL's version number as part of the directory and file name (e.g. sbl_ota_v1p20.bin is the update file for SBL v1.20). A reference JLink script for loading a SBL update is provided:

/tools/apollo510_scripts/jlink-ota-bootrom.txt

SBL updates can also be loaded using the configured wired interface (UART/SPI) in the same manor that other update images are. The existing SBL detects that it is a SBL update image and will pass it to the SBR for it to process and replace the current SBL.

When an SBL update is released there is also a corresponding MRAM Recovery image released that can be found in:

/tools/apollo510_scripts/mram_recovery/ambiq_recovery_images

SECTION 14

Appendix

14.1 INFOC-OTP Configuration

INFOC-OTP Region Start Address: 0x400C2000.

Table 14-1: INFOC-OTP Region Start Address: 0x400C2000

OTP Field	Purpose	OTP (word) Offset	MSB	LSB	Num Bits	Read Permissions	Write Permissions
RoT (HBK1)	128b Truncated hash of OEM Public Key	0x15	31	0	32	ALL	LCS=CM or LCS=DM
		0x16	31	0	32	ALL	LCS=CM or LCS=DM
		0x17	31	0	32	ALL	LCS=CM or LCS=DM
		0x18	31	0	32	ALL	LCS=CM or LCS=DM
KCP	128b OEM Symmetric Key used for Provisioning	0x19	31	0	32	LCS=CM or LCS=DM	LCS=DM or LCS=RMA (CM = 0's)
		0x1A	31	0	32	LCS=CM or LCS=DM	LCS=DM or LCS=RMA (CM = 0's)
		0x1B	31	0	32	LCS=CM or LCS=DM	LCS=DM or LCS=RMA (CM = 0's)
		0x1C	31	0	32	LCS=CM or LCS=DM	LCS=DM or LCS=RMA (CM = 0's)
KCE	128b OEM Symmetric Key used for Encryption	0x1D	31	0	32	LCS=CM or LCS=DM	LCS=DM or LCS=RMA (CM = 0's)
		0x1E	31	0	32	LCS=CM or LCS=DM	LCS=DM or LCS=RMA (CM = 0's)
		0x1F	31	0	32	LCS=CM or LCS=DM	LCS=DM or LCS=RMA (CM = 0's)
		0x20	31	0	32	LCS=CM or LCS=DM	LCS=DM or LCS=RMA (CM = 0's)
OEM-programmed flags (Key Attributes)	Number of "0" bits in HBK1	0x21	7	0	8	ALL	LCS=DM or LCS=RMA
	Number of "0" bits in KCP	0x21	14	8	7	LCS=CM or LCS=DM	LCS=DM or LCS=RMA
	KCP not in use	0x21	15	15	1	ALL	LCS=DM or LCS=RMA
	Number of "0" bits in KCE	0x21	22	16	7	LCS=CM or LCS=DM	LCS=DM or LCS=RMA
	KCE not in use	0x21	23	23	1	ALL	LCS=DM or LCS=RMA
	Reserved	0x21	29	24	6	ALL	ALL

Table 14-1: INFOC-OTP Region Start Address: 0x400C2000 *(Continued)*

OTP Field	Purpose	OTP (word) Offset	MSB	LSB	Num Bits	Read Permissions	Write Permissions
HBK1_MINVER	NV counter for Certificate Version (0-95)	0x24	31	0	32	ALL	ALL
		0x25	31	0	32	ALL	ALL
		0x26	31	0	32	NONE	ALL
SECURITY	Secure Boot Enable	0x27	10	8	3	Bootloader	DM | RMA
	Secondary Bootloader Enable	0x27	11	11	1	Bootloader	DM | RMA
	Disable INFO0 Programming	0x27	15	12	4	Bootloader	DM | RMA
	Enable Secureboot for Warm Reset	0x27	30	28	3	Bootloader	DM | RMA
	Ambiq Reserved (All other bits are not to be programmed)	0x27	--	--	--	Bootloader	DM | RMA
LOCKMASK	Debug Control Lock Enable	0x2A	31	0	32	ALL	LCS=CM or LCS=DM
		0x2B	31	0	32	ALL	LCS=CM or LCS=DM
SBL_WPROT	Write Protect MRAM sectors (16K granularity) - Upgradable by Ambiq SBL	0x80	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x81	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x82	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x83	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x84	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x85	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x86	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x87	31	0	32	Bootloader	LCS=DM or LCS=RMA
SBL_RPROT	Copy Protect MRAM sectors (16K granularity) - Upgradable by Ambiq SBL	0x88	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x89	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x8A	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x8B	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x8C	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x8D	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x8E	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x8F	31	0	32	Bootloader	LCS=DM or LCS=RMA
WPROT	Permanent Write Protect MRAM sectors (16K granularity) - Pre-installed one time programmed content	0x96	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x97	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x98	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x99	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x9A	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x9B	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x9C	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0x9D	31	0	32	Bootloader	LCS=DM or LCS=RMA

Table 14-1: INFOC-OTP Region Start Address: 0x400C2000 *(Continued)*

OTP Field	Purpose	OTP (word) Offset	MSB	LSB	Num Bits	Read Permissions	Write Permissions
RPROT	Permanent Copy Protect MRAM sectors (16K granularity)	0xA8	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0xA9	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0xAA	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0xAB	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0xAC	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0xAD	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0xAE	31	0	32	Bootloader	LCS=DM or LCS=RMA
		0xAF	31	0	32	Bootloader	LCS=DM or LCS=RMA
RSV	OEM General Purpose Space	0x91-0x92	31	29	3	~PROTLOCK	LCS=DM or LCS=RMA
SEC_POL	Enforce Authentication for Updates with CC	0x93	31	29	3	Bootloader	LCS=DM or LCS=RMA
	Enforce Encryption for Updates	0x93	28	26	3	Bootloader	LCS=DM or LCS=RMA
	Enforce Authentication for Updates	0x93	25	23	3	Bootloader	LCS=DM or LCS=RMA
	Reserved	0x93	22	8	15	Bootloader	LCS=DM or LCS=RMA
	SWO Pin	0x93	7	0	1	Bootloader	LCS=DM or LCS=RMA
BOOT_OVERRIDE	SecureBoot Override Config	0x94	31	0	32	Bootloader	LCS=DM or LCS=RMA
WIRED_CONFIG	Wired Interface Config	0x95	31	0	32	Bootloader	LCS=DM or LCS=RMA
CUSTOTP_READ_KEY	128b Secret Key used to unlock the Keybank for reading	0xA0	31	0	32	Ambiq SBR	LCS=DM or LCS=RMA
		0xA1	31	0	32	Ambiq SBR	LCS=DM or LCS=RMA
		0xA2	31	0	32	Ambiq SBR	LCS=DM or LCS=RMA
		0xA3	31	0	32	Ambiq SBR	LCS=DM or LCS=RMA
CUSTOTP_PROG_KEY	128b Secret Key used to unlock the 2nd half of Keybank for Writing	0xA4	31	0	32	Ambiq SBR	LCS=DM or LCS=RMA
		0xA5	31	0	32	Ambiq SBR	LCS=DM or LCS=RMA
		0xA6	31	0	32	Ambiq SBR	LCS=DM or LCS=RMA
		0xA7	31	0	32	Ambiq SBR	LCS=DM or LCS=RMA
CUSTOTP_PROGLOCK	Controls Write Access to 2nd Half of Keybank	0x9E	3	0	4	Ambiq SBR	LCS=DM or LCS=RMA
CUSTOTP_RDLOCK	Controls Read Access to Keybank	0x9F	7	0	8	Ambiq SBR	LCS=DM or LCS=RMA
KEYBANK_MFG	Keybank Q0	0xB0-0xBF	31	0	32	((~CUSTOTP_RDLOCK[0] & (Bootloader)) | (~CUSTOTP_RDLOCK[4] & CUST_KEYBANK_KEY)) & LCS!=RMA	LCS=DM or LCS=RMA

Table 14-1: INFOC-OTP Region Start Address: 0x400C2000 *(Continued)*

OTP Field	Purpose	OTP (word) Offset	MSB	LSB	Num Bits	Read Permissions	Write Permissions
KEYBANK_MFG	Keybank Q1	0xC0-0xCF	31	0	32	((~CUSTOTP_RDLOCK[1] & (Bootloader)) | (~CUSTOTP_RDLOCK[5] & CUST_KEYBANK_KEY)) & LCS!=RMA	LCS=DM or LCS=RMA
KEYBANK_RT	Keybank Q2 (Can be programmed runtime)	0xD0-0xDF	31	0	32	((~CUSTOTP_RDLOCK[2] & (Bootloader)) | (~CUSTOTP_RDLOCK[6] & CUST_KEYBANK_KEY)) & LCS!=RMA	LCS=DM or LCS=RMA or ((~CUSTOTP_PROGLOCK[0] & (Bootloader)) | (~CUSTOTP_PROGLOCK[2] & CUSTOTP_PROG_KEY))
KEYBANK_RT	Keybank Q3 (Can be programmed runtime)	0xE0-0xEF	31	0	32	((~CUSTOTP_RDLOCK[3] & (Bootloader)) | (~CUSTOTP_RDLOCK[7] & CUST_KEYBANK_KEY)) & LCS!=RMA	LCS=DM or LCS=RMA or ((~CUSTOTP_PROGLOCK[1] & (Bootloader)) | (~CUSTOTP_PROGLOCK[3] & CUSTOTP_PROG_KEY))

6500 River Place Boulevard, Building 7, Suite 200, Austin, TX 78730
www.ambiq.com
sales@ambiq.com
+1 512. 879.2850

A-SOCAP5-UGGA04EN v1.0
May 2025