

# **AWR1642 Bootloader Flow**

Naveen N.

## **ABSTRACT**

This application report describes the AWR1642 bootloader flow.

### **Contents**

1	Introduction .....	2
2	Basic Bootloader Flow .....	5
3	Security Enhancements in Bootloader .....	9
4	Programming Serial Data Flash Over UART (Bootloader Service).....	13

### **List of Figures**

1	Simplified Representation of AWR1642 Interconnect.....	2
2	Flashing Mode of Bootloader.....	3
3	Execution Mode of Bootloader.....	4
4	Basic Bootloader Flow Chart.....	5
5	Image Load Sequence .....	6
6	ROM-Assisted Image Download Sequence.....	7
7	Bootmode – SPI .....	8
8	Hardware Infrastructure Used for Boot Security.....	9
9	Key Management .....	10
10	SDF Image Layout .....	11
11	Secure Boot Flow .....	12
12	Host ← → AWR Device UART Communication .....	15
13	Flashing Sequence.....	16

### **List of Tables**

1	SOP Lines and Boot Modes.....	2
2	Supported Commands and Format .....	14

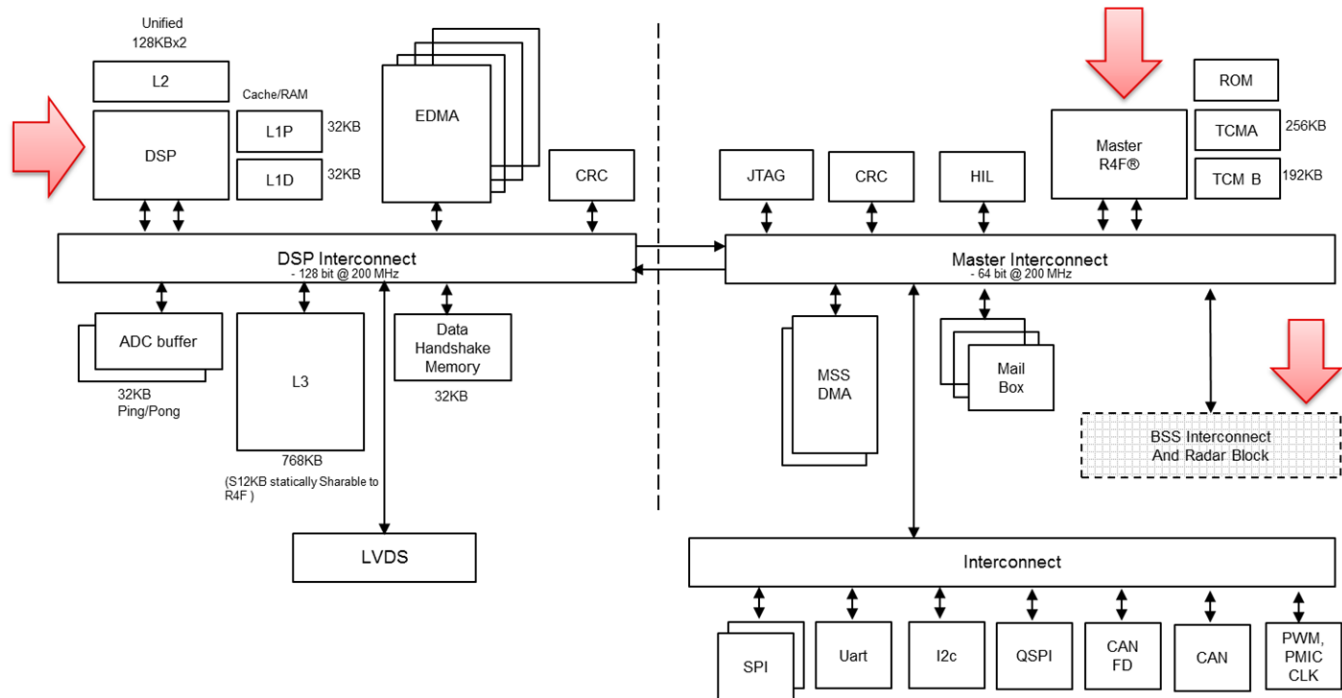
## **Trademarks**

ARM, Cortex are registered trademarks of ARM Limited.  
 Macronix is a registered trademark of Macronix International Co., Ltd.  
 Spansion is a registered trademark of Spansion LLC.  
 All other trademarks are the property of their respective owners.

# 1 Introduction

The AWR1642 device can be broadly split into three subsystems (see Figure 1), as follows:

- Master subsystem: ARM® Cortex®-R4F and associated peripherals, hosts the user application
- DSP subsystem: TI C674x and associated peripherals, hosts the user application
- Radar/Millimetre Wave Block: Programmed using predefined message transactions specified by TI (reference driver provided by TI)



**Figure 1. Simplified Representation of AWR1642 Interconnect**

User application components (R4F and DSP) are expected to be stored in the serial data flash (SDF) interfaced to the AWR1642 device over the quad serial peripheral interface (QSPI) interface.

Master subsystem is the first programmable block to get activated after the AWR1642 device reset is deasserted. The bootloader of the AWR1642 device is hosted in the read-only memory (ROM) of the master subsystem, and takes control immediately.

From this point onward, the AWR1642 bootloader can operate in two modes: flashing and execution

The bootloader checks the state of the sense on power (SOP) I/Os – SOP lines driven externally for choosing the specific mode (see Table 1).

**Table 1. SOP Lines and Boot Modes**

SOP2 (P13)	SOP1 (P11)	SOP0 (J13)	Bootloader Mode and Operation
0	0	1	Functional mode The device bootloader loads the user application from the QSPI serial flash to the internal RAM and switches the control to it.
1	0	1	Flashing mode The device bootloader spins in loop to allow flashing of the user application (or the device firmware patch – supplied by TI).
0	1	1	Debug mode The bootloader is bypassed and the R4F processor is halted. This lets the user connect the emulator at a known point.

Flashing mode of the bootloader allows an external entity to load the customer application image to the SDF (see Figure 2).

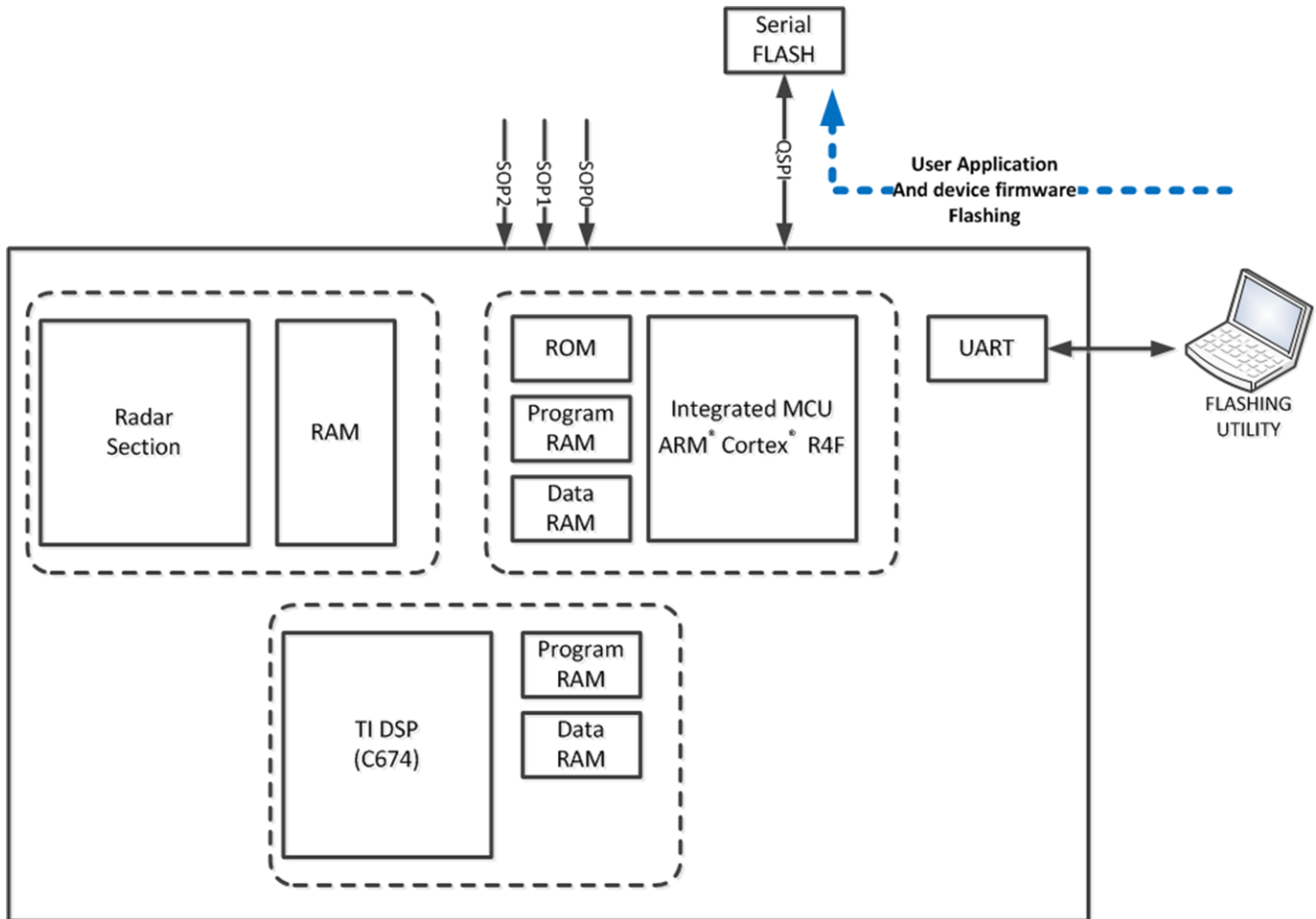
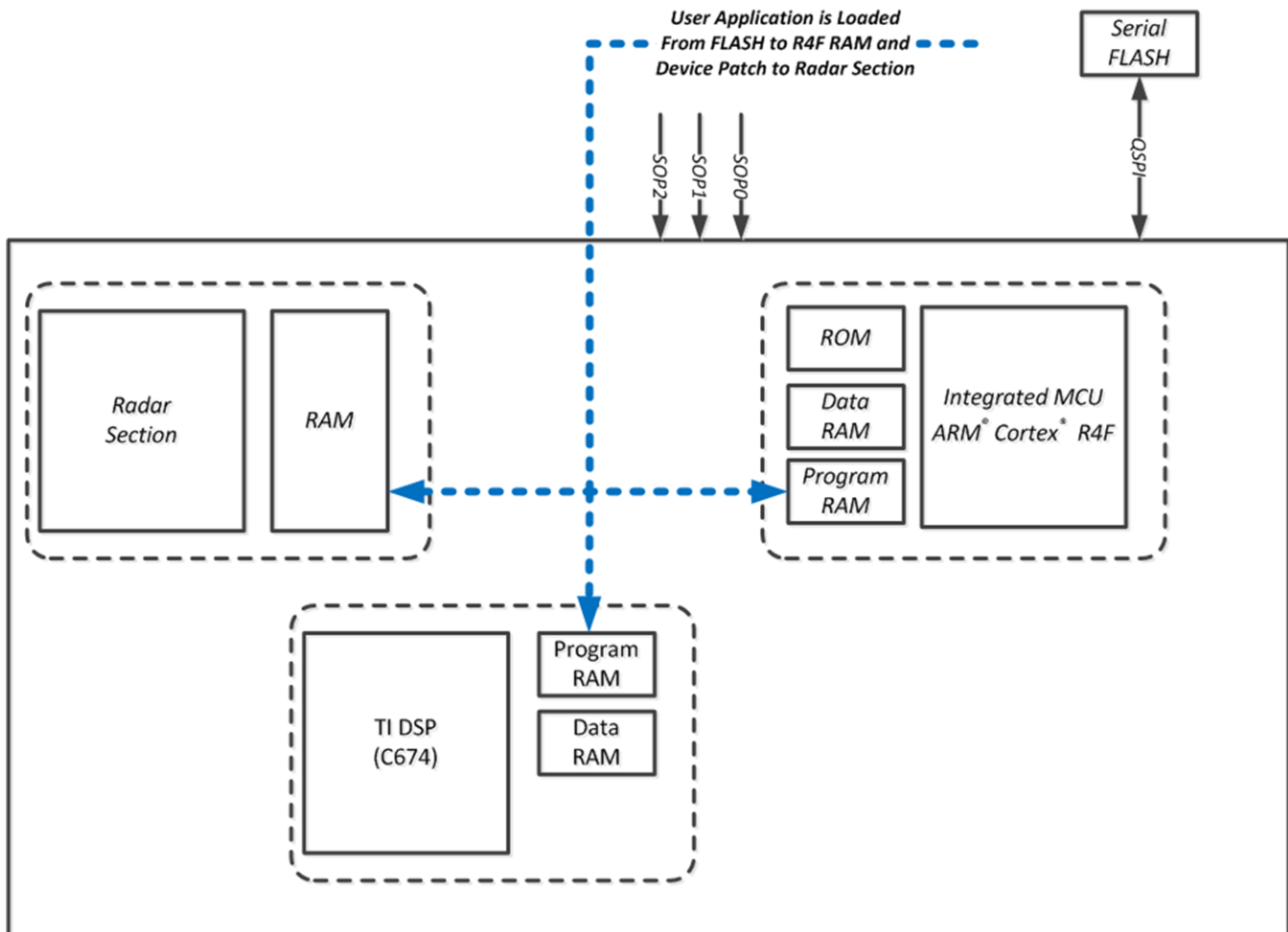


Figure 2. Flashing Mode of Bootloader

Execution (or functional) mode of the bootloader relocates the image stored in the SDF to the R4F and DSP memory subsystems. Toward the end of this process, the bootloader passes the R4F application of the control user. Unhalting (starting execution) of the DSP core is the responsibility of the user image (see Figure 3).



**Figure 3. Execution Mode of Bootloader**

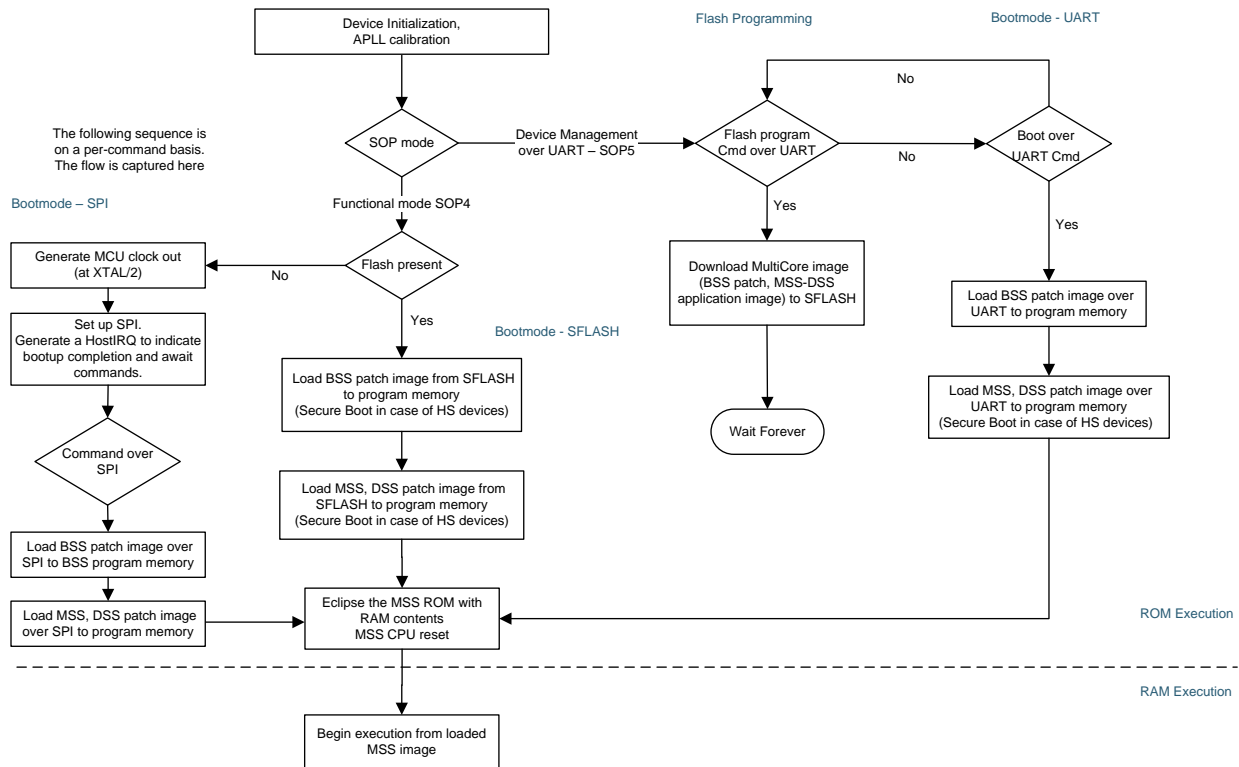
#### Key points

- TI's embedded bootloader can load one primary user image (could have content for both R4F and DSP).
- If the customer application requires handling of multiple images (factory programmed, back-up, and so on), the customer must invest in a secondary bootloader.

## 2 Basic Bootloader Flow

At a high level, bootloader operation can be split into three phases (see Figure 4), as follows:

- Device initialization: the bootloader uses built-in self test (BIST) engines for hardware diagnostics (for example, RAM tests).
- Setting up the root clock by starting the APLL. The root clock will be at 200 MHz.
- Checking SOP lines to proceed with either the flashing or execution mode.



**Figure 4. Basic Bootloader Flow Chart**

### Key points

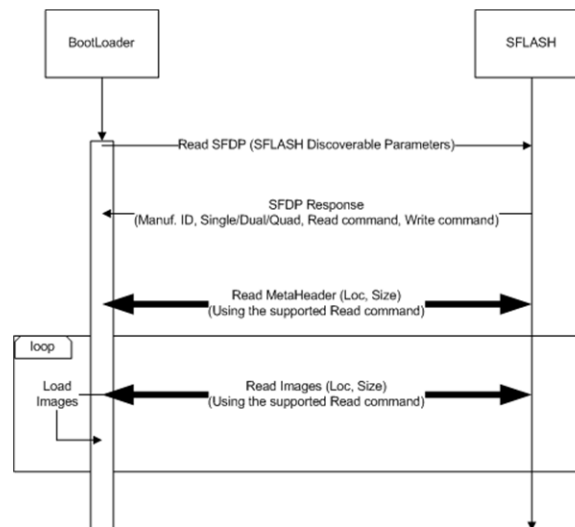
- In addition to the memories of the Radar subsystem, the bootloader loads to the following memories:
  - MSS images – MSS TCMA and MSS TCMB (on AWR16xx ES1.0 samples, the load is restricted to MSS TCMA program memory)
  - DSP images – L1, L2, and L3 memories (on AWR16xx ES1.0 samples, the load is restricted to L2 and L3 memories)

## 2.1 Bootmode – SFLASH

### 2.1.1 Image Load Sequence

In functional mode, the bootloading of an image from the SDF is the first bootmode attempted by the bootloader (see [Figure 5](#)). This bootmode involves the following steps:

1. Pinmux the QSPI pins of the AWR1642 device:
  - [QSPI[0]: Ball R13
  - QSPI[1]: Ball N12
  - QSPI[2]: Ball R14
  - QSPI[3]: Ball P12
  - QSPI\_CLK: Ball R12
  - QSPI\_CS\_N: Ball P11
2. QSPI is set up to operate at  $(\text{system clock} / 5) = (200/5) = 40$  MHz.
3. The SFLASH discoverable parameters (SFDP) command is issued to retrieve the JEDEC compliant response, which includes information regarding the SFLASH capabilities and command set. When the SFDP response is received, the information is used to communicate with the SDF and further interpret the contents and load the images.



**Figure 5. Image Load Sequence**

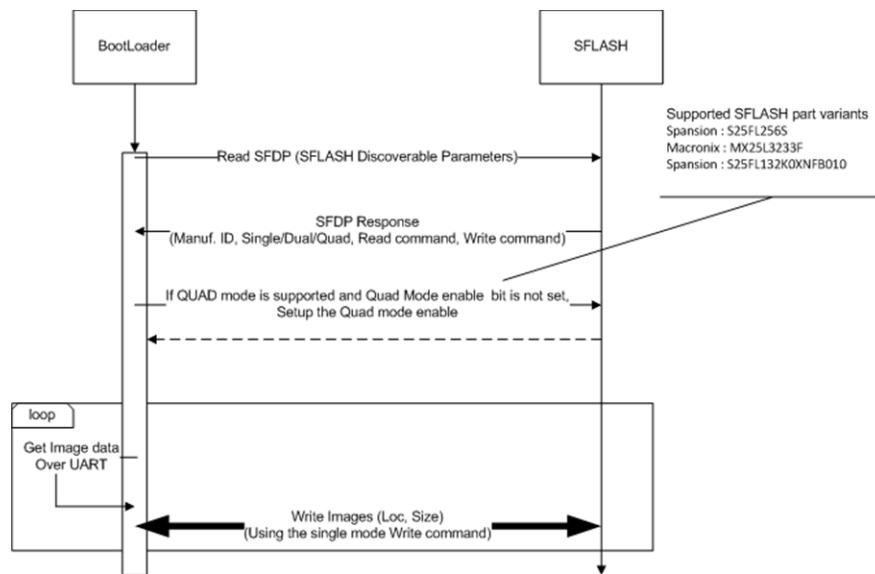
#### Key points

- The ROM bootloader performs the read from the SDF, based on the highest capability mode (quad, dual, or single) as published by the SDF in response to the SFDP command.
- For SDF variants that support quad mode, the quad mode commands are issued; if the quad enable (QE) bit is not set, the communication will fail. In such cases, the load flow assumes that the QE bit in the SDF is already set.
- Fallback images: the bootloader supports loading of images from the following locations as a fallback mechanism if one of the images is corrupted in the SDF. The locations of the images are:
  - META IMG1(SDF offset – 0x0)
  - META IMG2(SDF offset – 0x80000)
  - META IMG3(SDF offset – 0x100000)
  - META IMG4(SDF offset – 0x180000)

See the Image Creator user guide available in the mmWave SDK release for image format details.

### 2.1.2 ROM-Assisted Image Download Sequence

The ROM-assisted image download sequence is entered by placing the device in flashing mode. See [Section 4](#), for further details on the handshake with an external host to receive the image. [Figure 6](#) shows the communication with the SDF.



**Figure 6. ROM-Assisted Image Download Sequence**

#### Key points

- The ROM-assisted download should work with all flash variants that allow for *memory-mapped mode* and *Page program command (0x2)*, with one dummy byte and 24-bit addressing.
- Setting the QE bit varies from one SDF vendor to another. The ROM bootloader supports setting the QE bit for Spanion® and Macronix® variants (certain specific part variants only) in this flow.
- In addition to a checksum-based integrity check for every packet received over the UART, a CRC32-based integrity check is performed over the complete image. The CRC32 is computed incrementally as the packets are received and written to the SDF.

### 2.2 Bootmode – SPI

In functional mode, if and only if the detection of the SDF fails (concluded by an invalid response to the SFDP command over the QSPI lines), the bootloader enters the SPI-based bootloading mode. This mode involves the following steps:

1. Pinmux the SPI pins of the AWR1642 device:
  - SPI\_MOSI: Ball D13
  - SPI\_MISO: Ball E14
  - SPI\_CLK: Ball E13
  - SPI\_CS\_N: Ball C13
  - SPI\_HOST\_INTR: Ball P13
2. Follow the Communication Protocol as defined in the Radar Interface Control document, to communicate with an external host to receive the images to be loaded as message packets over the SPI.
3. Once the loading of all images is complete, the ROM is eclipsed and execution control is transferred to the loaded application in MSS TCMA.

Figure 7 shows the handshake with the external host.

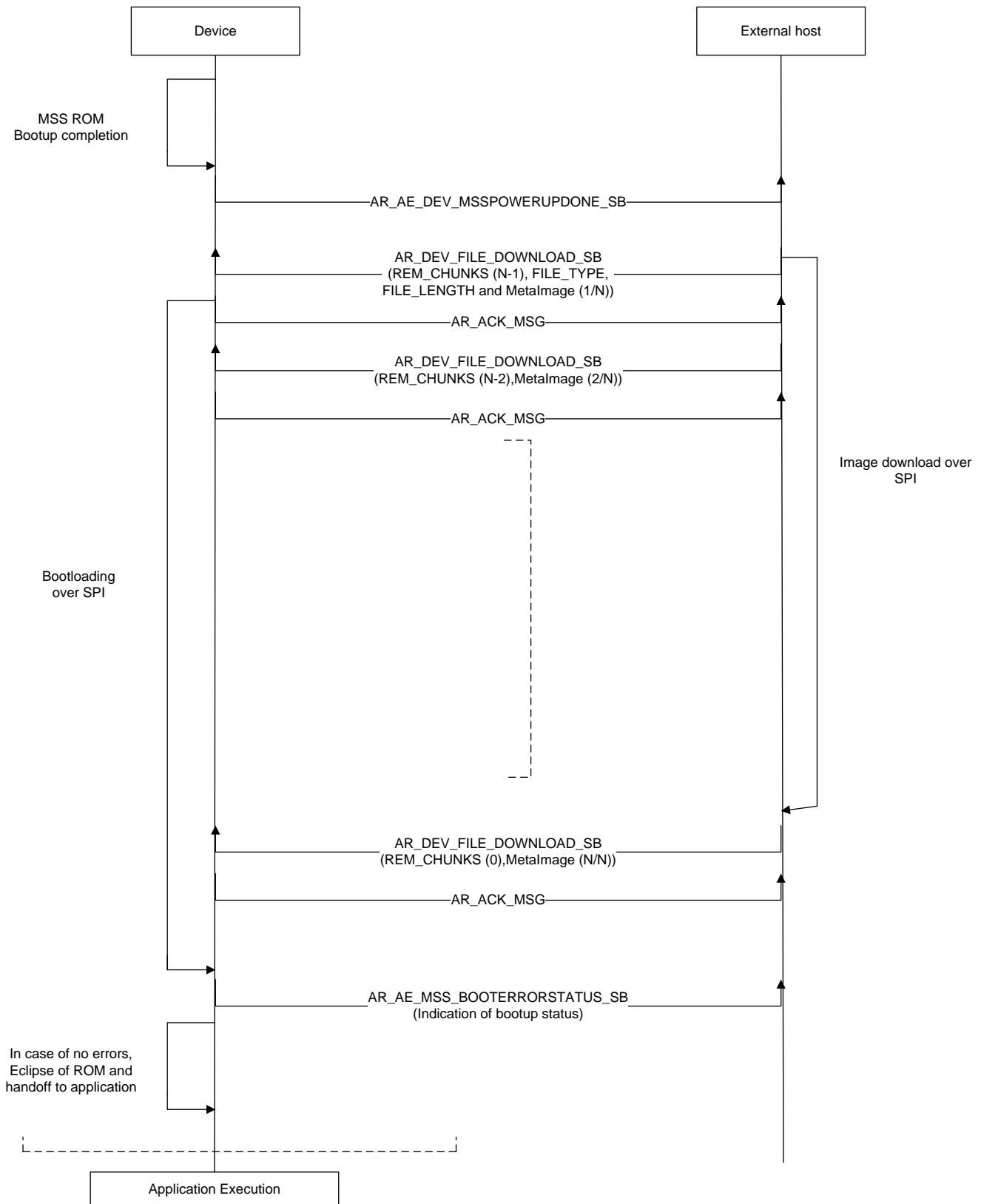


Figure 7. Bootmode – SPI



### 3 Security Enhancements in Bootloader

The AWR1642 device is also available with security enhancements (referred to as high security or HS device). One of the key features of the HS device is secure boot. At a high level, this feature attends to the two following capabilities:

- Take over protection: support for asymmetric-key-based authentication. The bootloader has the capability of authenticating an image signed with a customer key.
- IP Protection: symmetric-key-based encryption and decryption support. The customer image is not kept as plain text in the SDF.

#### 3.1 Hardware Infrastructure

The key hardware enabler in the HS device (see Figure 8) is the capability of the device to do the following:

- Allow burning of customer keys
- Provide hardware accelerators for cryptographic operations (AES, public key accelerator, and more)

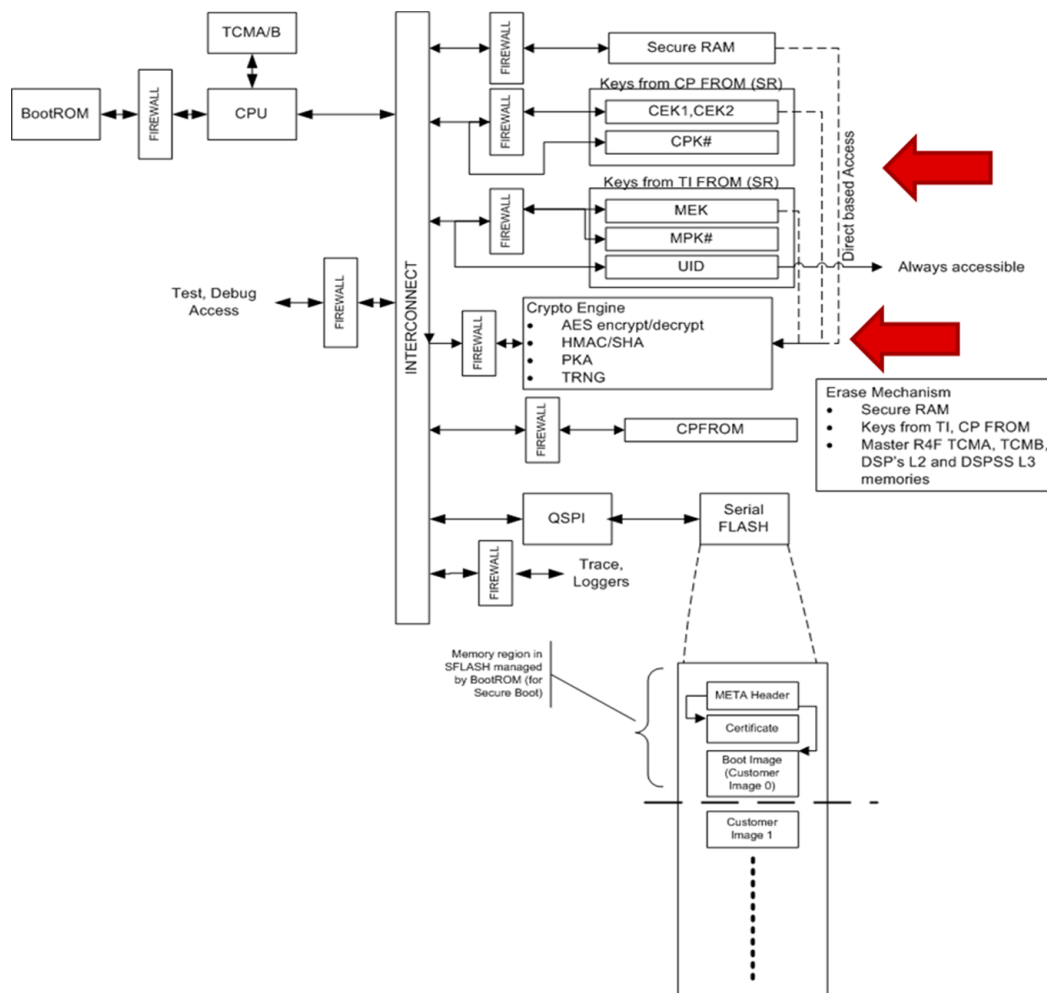


Figure 8. Hardware Infrastructure Used for Boot Security

### 3.2 Secure Boot

Before exploring the secure boot flow, two important concepts must be introduced, as follows.

#### 3.2.1 Key Management

Figure 9 shows the key management concept.

1. HS devices are shipped with preburnt TI keys.
2. TI provides an embedded tool (signed with a TI key) to customers to burn their keys.
3. When the keys are burned, customers can create their image (see [Section 3.2.2](#)) using an image creation tool (provided as a reference by TI).
4. During this process, the customer image would be signed (and encrypted if required).
5. Using the flashing tool from TI (provided as reference), this image would be burnt on the SDF using flashing mode of the bootloader (transfer is over UART).

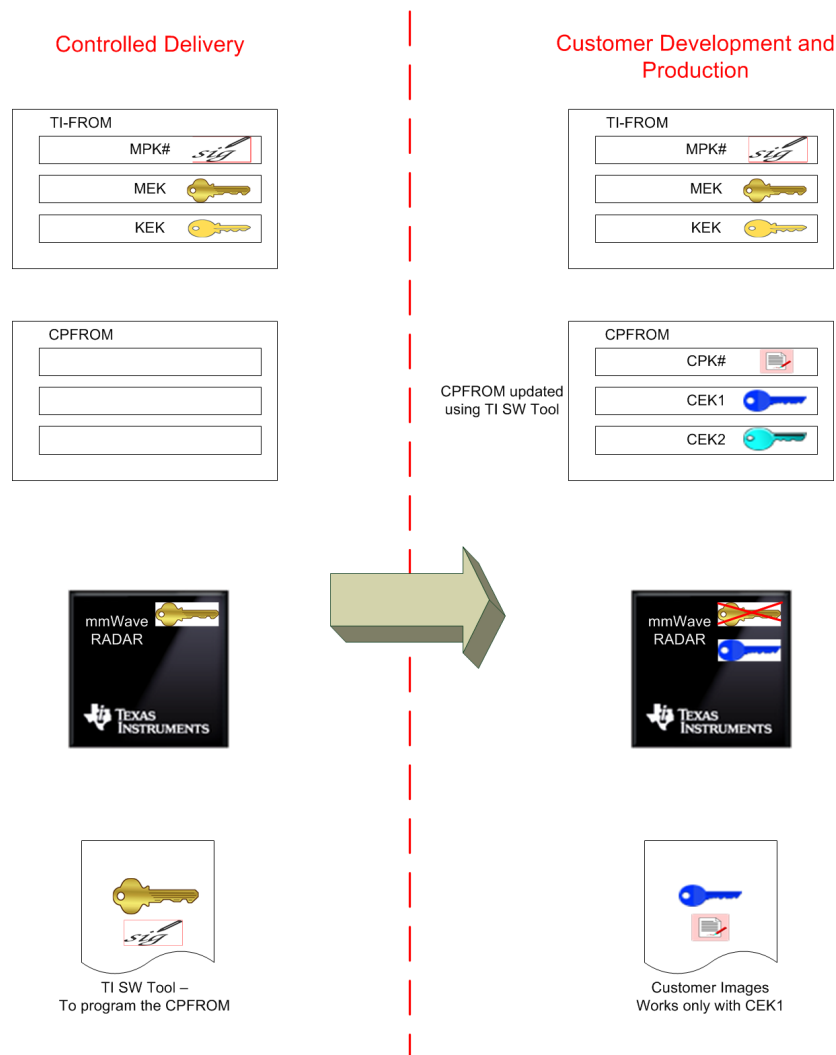


Figure 9. Key Management

### 3.2.2 Secure Image

Figure 10 shows the reference layout of the secure image as stored in the SDF.

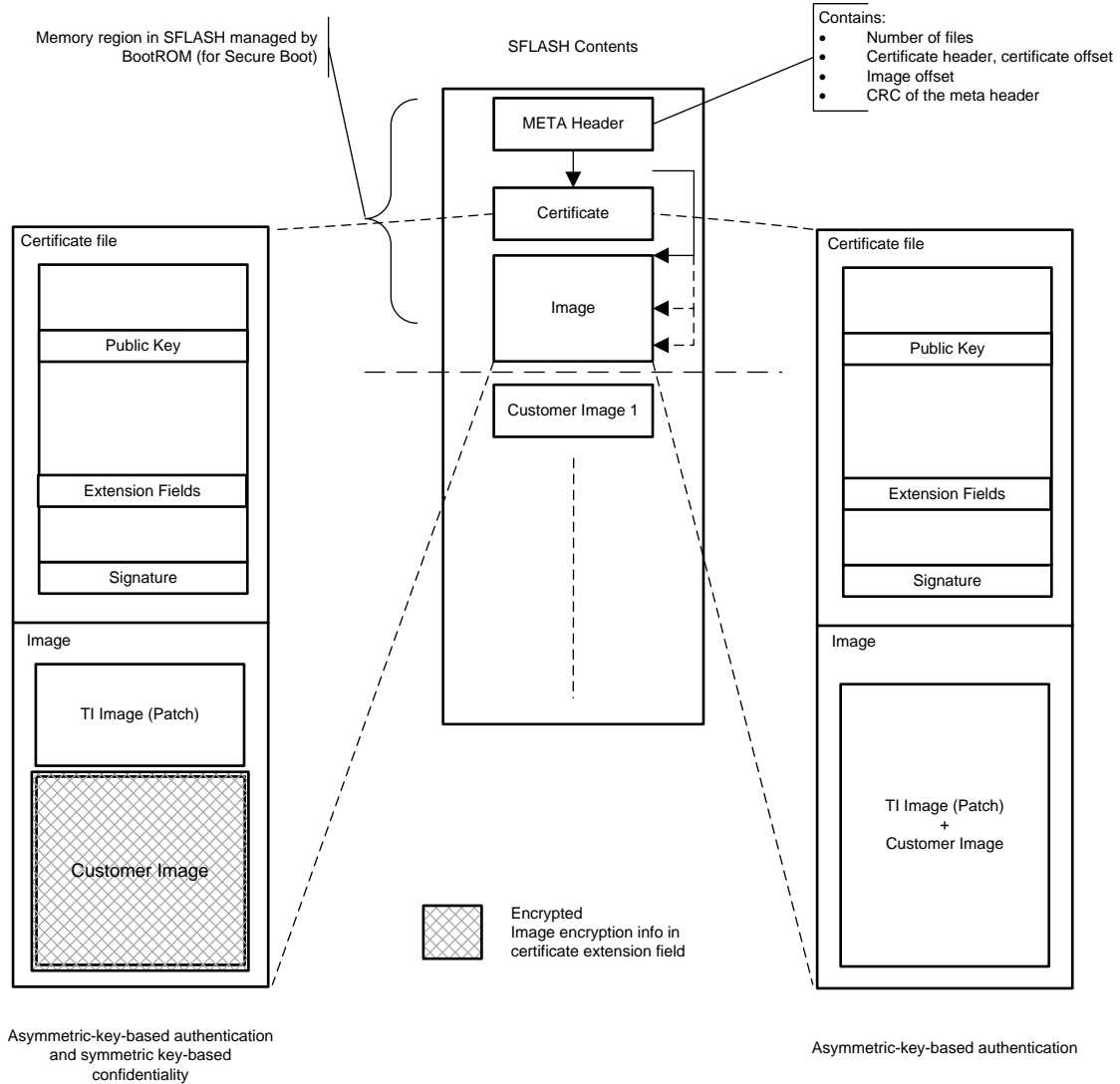
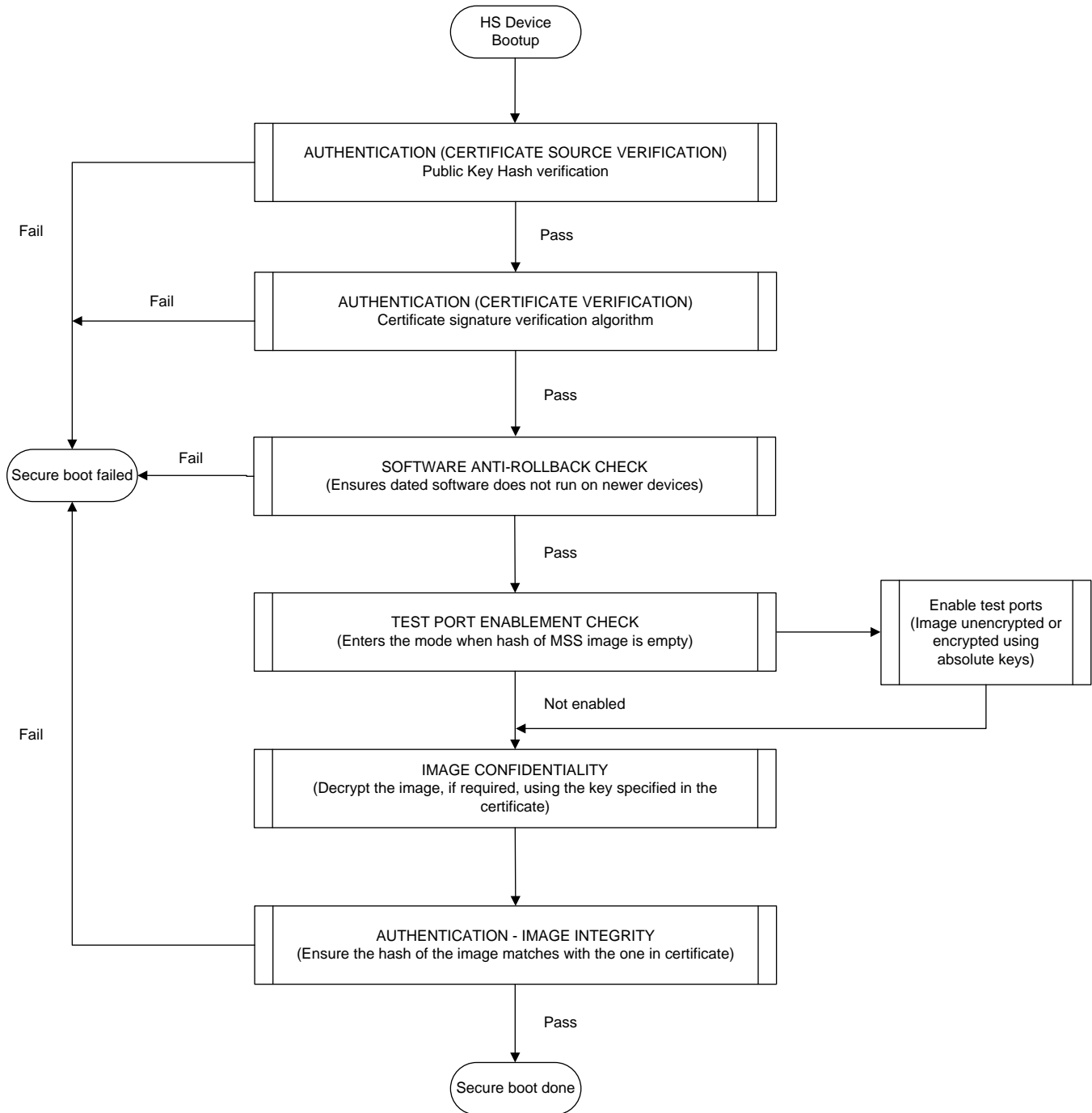


Figure 10. SDF Image Layout

Part of the SDF that is managed by the ROM bootloader is determined by the image header. The primary image would consist of customer application components (R4F and DSP), and possibly with a TI-provided ROM patch for the Radar Block along with a certificate. The rest of the SDF could be managed by the customer application.

### 3.2.3 Secure Boot Flow

Figure 11 shows the secure boot flow at a high level.



**Figure 11. Secure Boot Flow**

### 3.3 Note on Field Returns and Debugging With the HS Device

- AWR1642 HS devices are shipped with all debug interfaces (namely JTAG and Trace) disabled.
- The hardware has the capability to provide options to the authenticated user application to enable JTAG. This is the responsibility of the customer.
- If such a device, or rather a printed-circuit board with an HS device, is received by TI for debugging, the standard flow is to deploy a *test mode* (selected by SOP lines), where before opening the debug interfaces, SDF and RAM instances are completely erased (to protect the customer IP).

## 4 Programming Serial Data Flash Over UART (Bootloader Service)

The AWR1642 device from TI can be configured to operate as an autonomous radar sensor. In this configuration, the user application and TI firmware patches are hosted in an SDF interfaced to the AWR1642 over the QSPI port.

SDF programming supports downloading meta images that are a combination of four components, as follows:

- User application image for R4F (master subsystem)
- User application image for C674 (DSP subsystem)
- TI Radar Block patches
- Certificates in DER format; these apply to HS devices

The flash programmer connects to the device over UART. Specifics are as follows:

- MSS\_UARTA of the AWR1642 device:
  - RX: Ball N4
  - TX: Ball N5
- Baud rate: 115200
- Packet size: 256 bytes

### 4.1 Binary File Format

The target binary file is composed of the following sections:

- Header
- Certificate
- R4F application
- DSP application
- TI Radar Block patch

The mmWave SDK package for the AWR1642 device from TI includes the *Image Creator* utility, which constructs the complete image with the previously listed components.

### 4.2 Flash Programming Sequence

1. Boot the device in SOP 5 mode (see [Table 1](#)).
2. Open the *UniFlash* tool (as listed in the mmWave SDK for AWR1642).
3. Connect to the device over the UARTA com port (the device expects a UART break signal – this is generated by the UniFlash tool).
4. Flash the desired images <META\_IMAGE1/ META\_IMAGE2/ META\_IMAGE3/ META\_IMAGE4>

### 4.3 Supported Commands and Format

Table 2 lists the supported commands and format.

**Table 2. Supported Commands and Format**

Command	Command ID	Description	Fields
PING	0x20	The device responds with ACK	
OPEN FILE	0x21	Command that gives details about the type of file being downloaded	File size: total file size being downloaded. File type: META IMG1(4), META IMG2(5), META IMG3(6), and META IMG4(7)
WRITE FILE to SFLASH	0x24	Command that gives the content of the file to write to SFLASH	
WRITE FILE to RAM	0x26	Command that gives the content of the file and the file is directly written to RAM	
CLOSE FILE	0x22	Command that indicates the end-of-file download	File type: META IMG1(4), META IMG2(5), META IMG3(6), and META IMG4(7)
GET STATUS	0x23	Command that requests the status of the previous command. The device responds with the status of the previous command issued.	
ERASE DEVICE	0x28	Command to erase the contents of the SFALSH	
GET VERSION	0x2F	Command that requests the version of the ROM. Device responds with the version information.	
ACK response	0xCC	Response from the device	

Figure 12 the supported commands that can be issued to the AWR device during the flash programming process and the various responses from the AWR device.

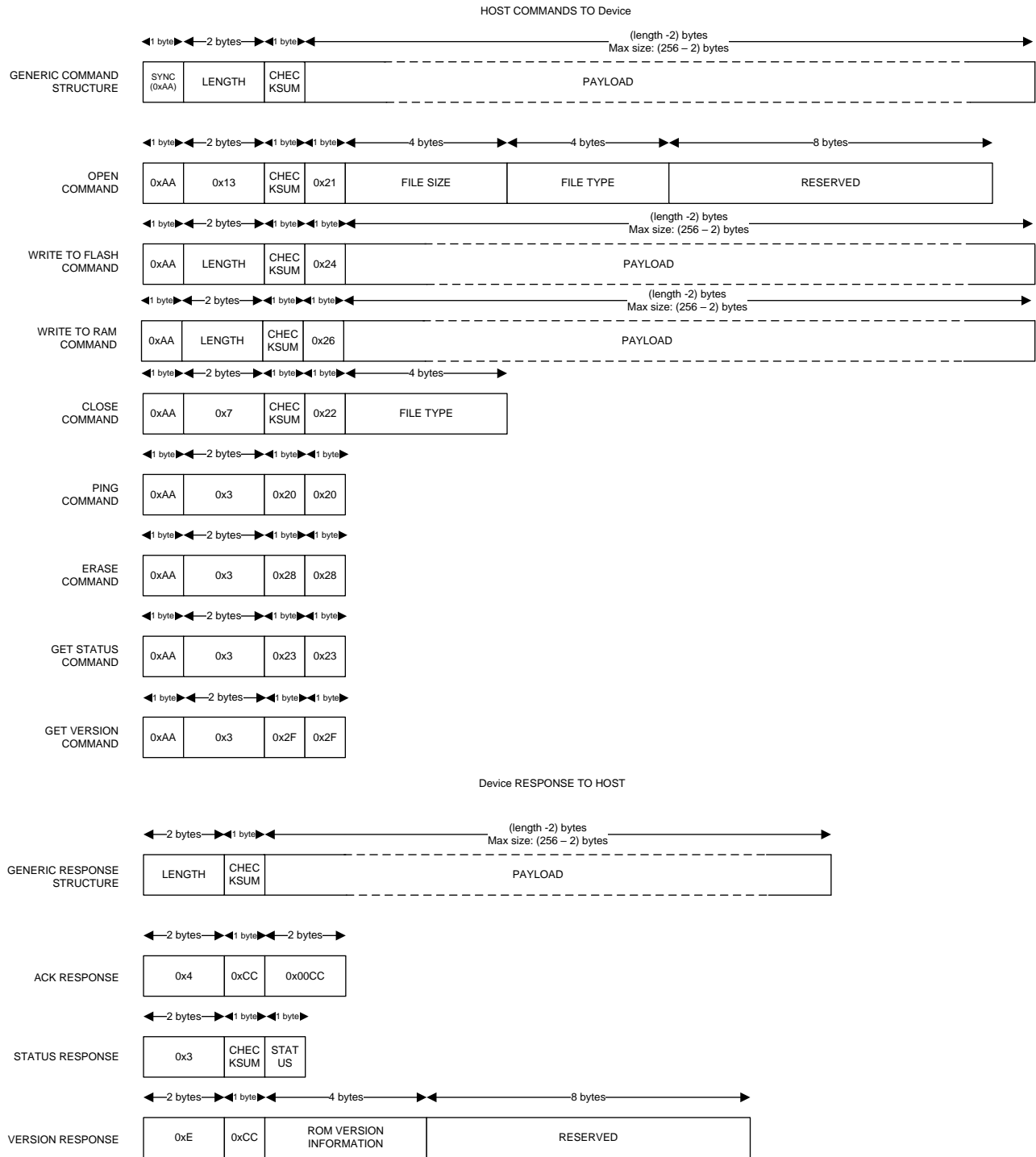
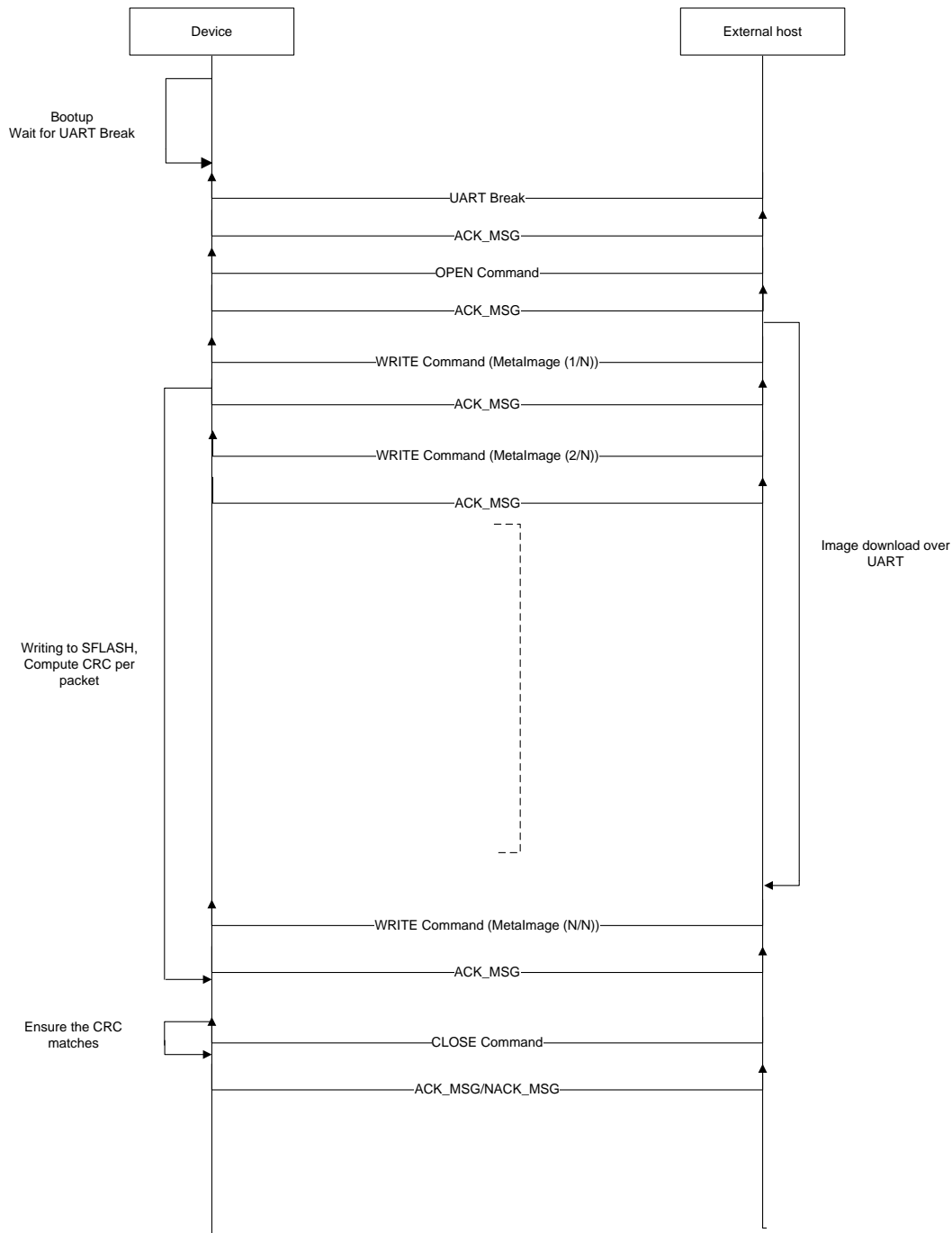


Figure 12. Host ← → AWR Device UART Communication

## 4.4 Flashing Sequence

Figure 13 shows the flash programming sequence. The initial handshake starts with a UART break issued by the external host. This break is followed by the command sequence in Figure 13.



**Figure 13. Flashing Sequence**

**Bootmode – UART:** The bootloading over the UART also follows the same sequence as previously mentioned (WRITE command – 0x26). The META IMAGE received over the UART is interpreted and loaded to the appropriate memories. Once the bootloading is complete, the ROM is eclipsed and execution control is passed to the application residing in MSS TCMA. The META IMAGE should not have the CRC32 appended (unlike the image to be flashed).



## IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ("TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products (<http://www.ti.com/sc/docs/stdterms.htm>), [evaluation modules](#), and [samples](http://www.ti.com/sc/docs/sampterm.htm) (<http://www.ti.com/sc/docs/sampterm.htm>).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265  
Copyright © 2017, Texas Instruments Incorporated