Application Report
SWRA563–June 2017

IWR1642 Bootloader Flow

Naveen N.

ABSTRACT

This application report describes the IWR1642 bootloader flow.

Contents

1	Introduction	2
2	Basic Bootloader Flow	5
3	Programming Serial Data Flash Over UART (Bootloader Service)	9

List of Figures

1	Simplified Representation of IWR1642 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	Host ← → IWR Device UART Communication	11
9	Flashing Sequence	12

List of Tables

1	SOP Lines and Boot Modes	2
2	Supported Commands and Format	10

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 IWR1642 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)

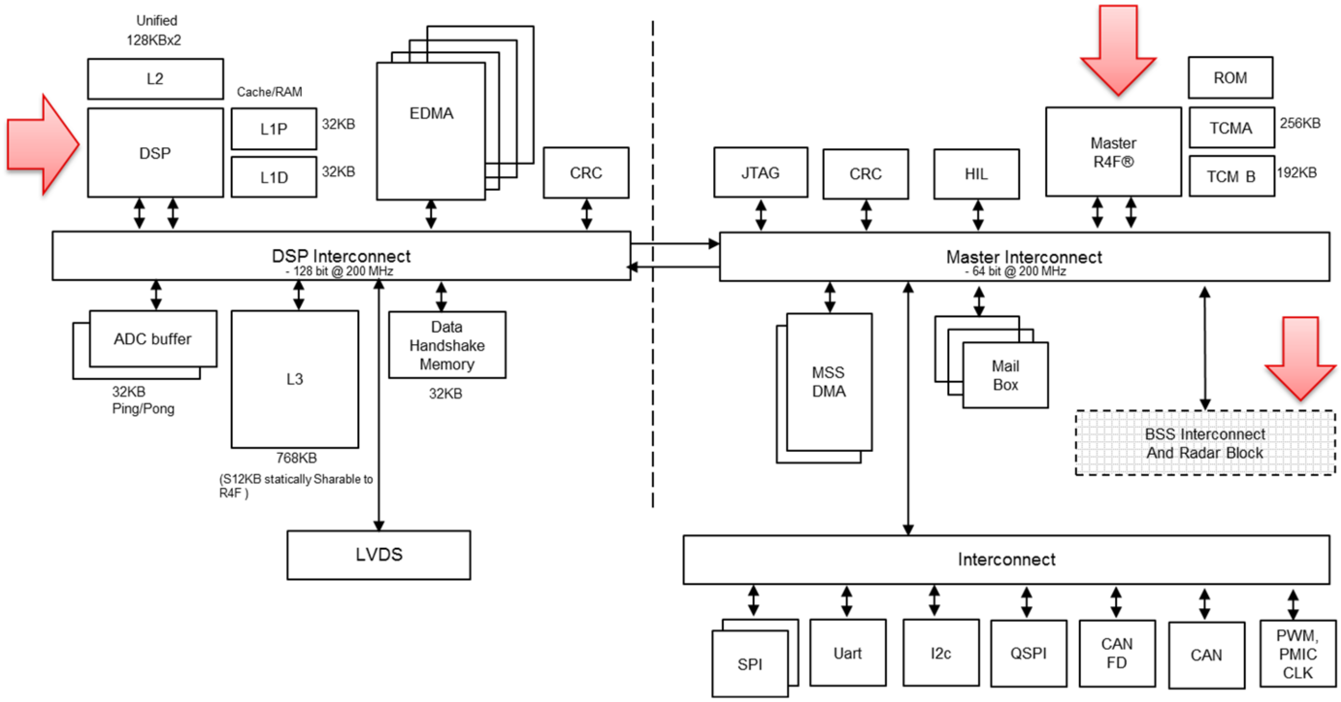

Figure 1. Simplified Representation of IWR1642 Interconnect

User application components (R4F and DSP) are expected to be stored in the serial data flash (SDF) interfaced to the IWR1642 device over the quad serial peripheral interface (QSPI) interface.

Master subsystem is the first programmable block to get activated after the IWR1642 device reset is deasserted. The bootloader of the IWR1642 device is hosted in the read-only memory (ROM) of the master subsystem, and takes control immediately.

From this point onward, the IWR1642 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).

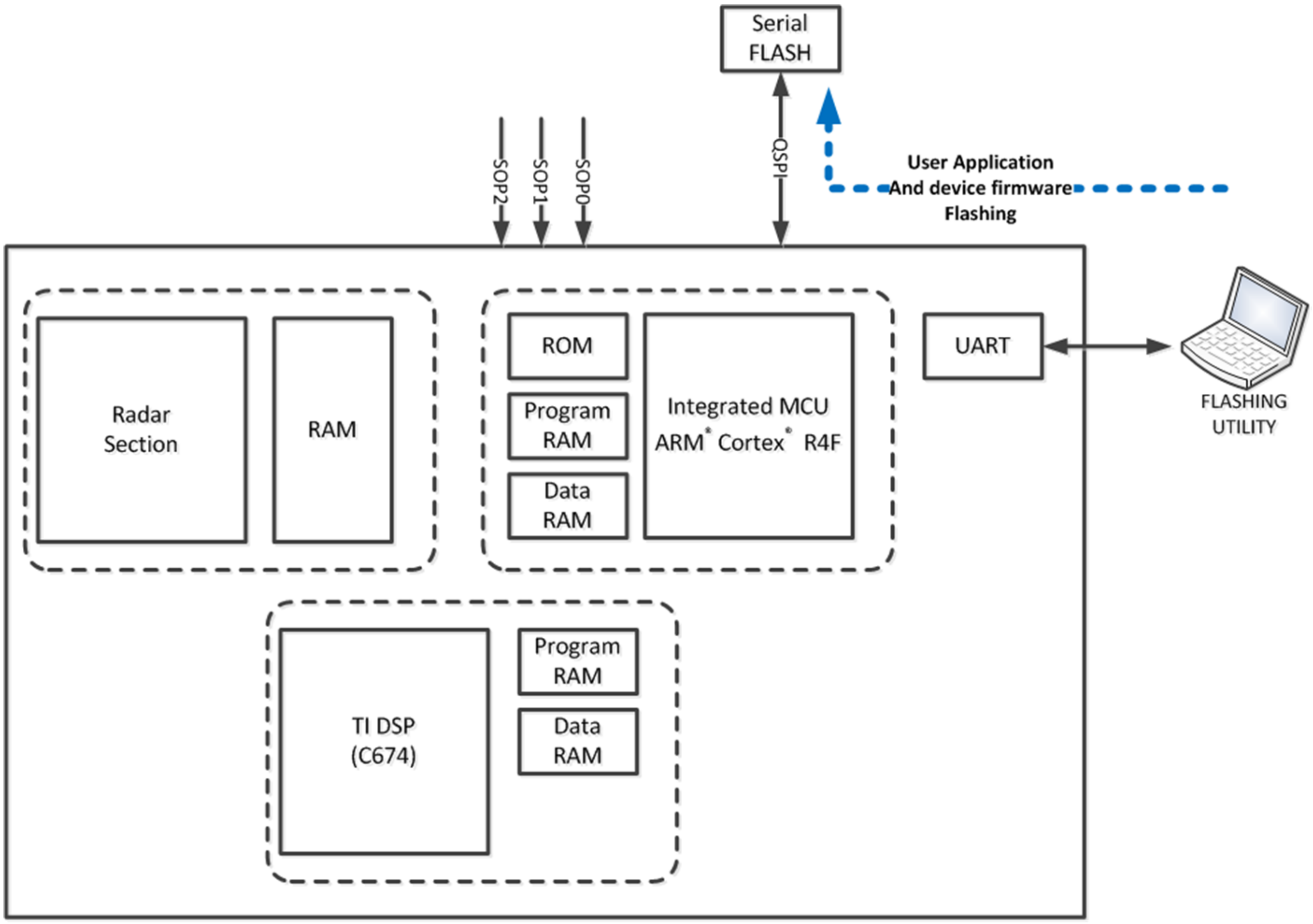

Figure 2. Flashing Mode of Bootloader

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).

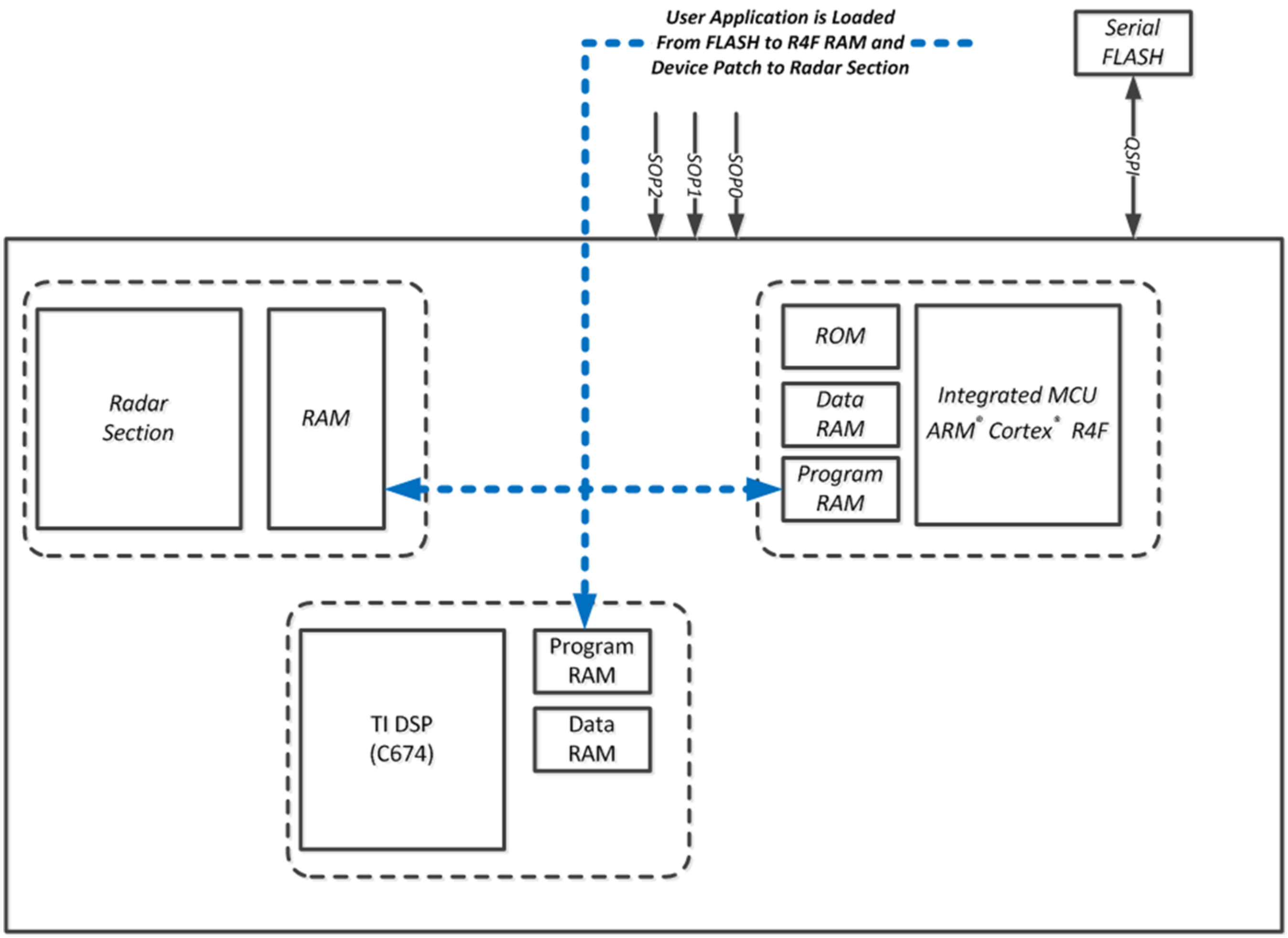

Figure 3. Execution Mode of Bootloader

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.

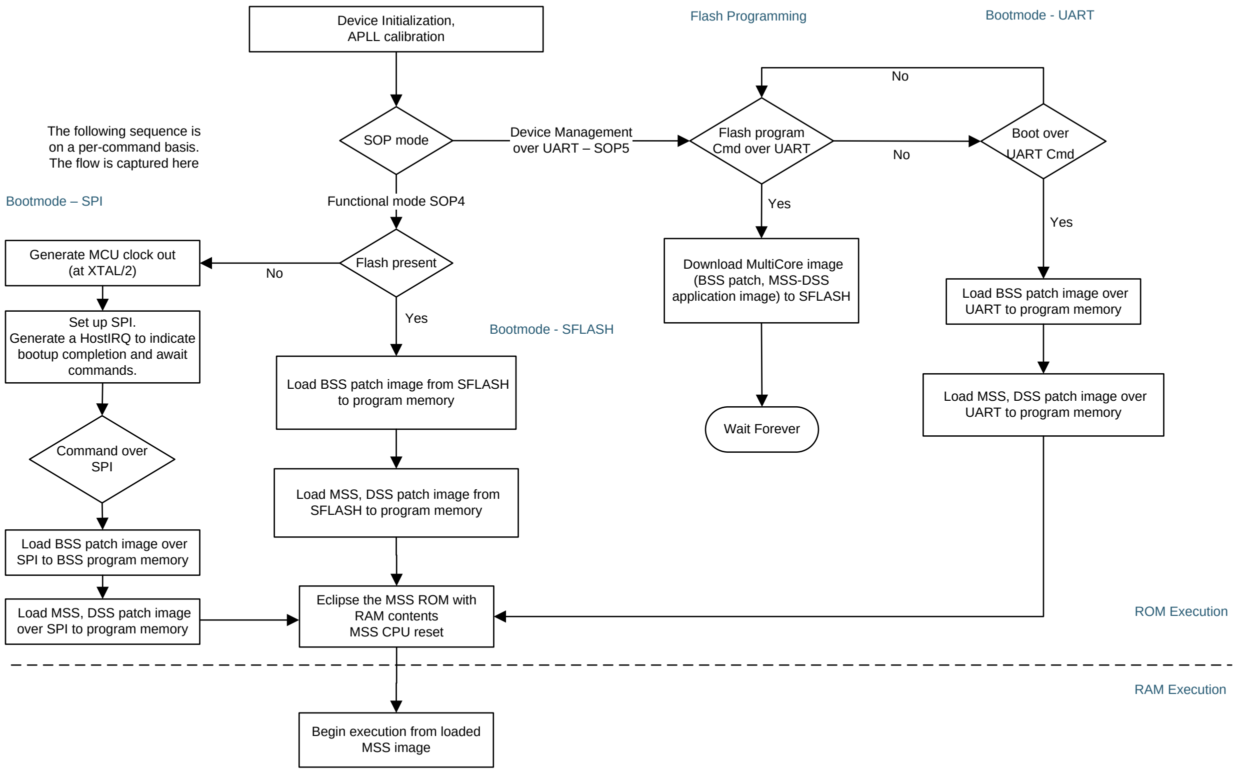

Figure 4. Basic Bootloader Flow Chart

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 IWR16xx ES1.0 samples, the load is restricted to MSS TCMA program memory)
  - DSP images – L1, L2, and L3 memories (on IWR16xx 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 IWR1642 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 (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.

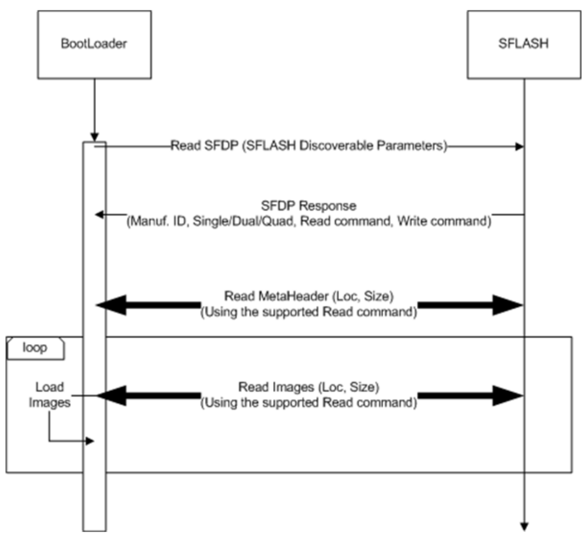

Figure 5. Image Load Sequence

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 3, for further details on the handshake with an external host to receive the image. Figure 6 shows the communication with the SDF.

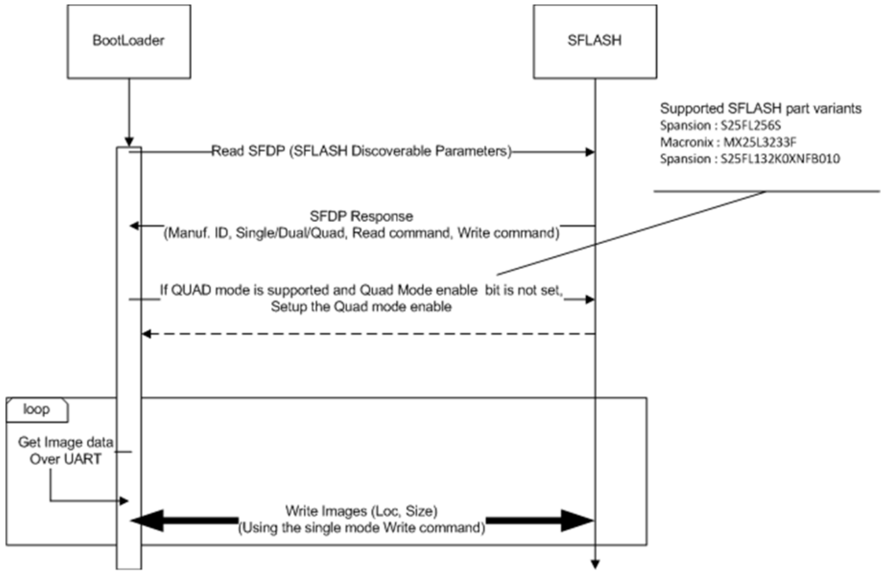

Figure 6. ROM-Assisted Image Download Sequence

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 Spansion® 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 IWR1642device:
   - 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 used by the mmWave SDK mmWaveLink rlDeviceFileDownload API (described in the mmWaveLink API Doxygen documentation referenced in the mmWave SDK User Guide) 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.

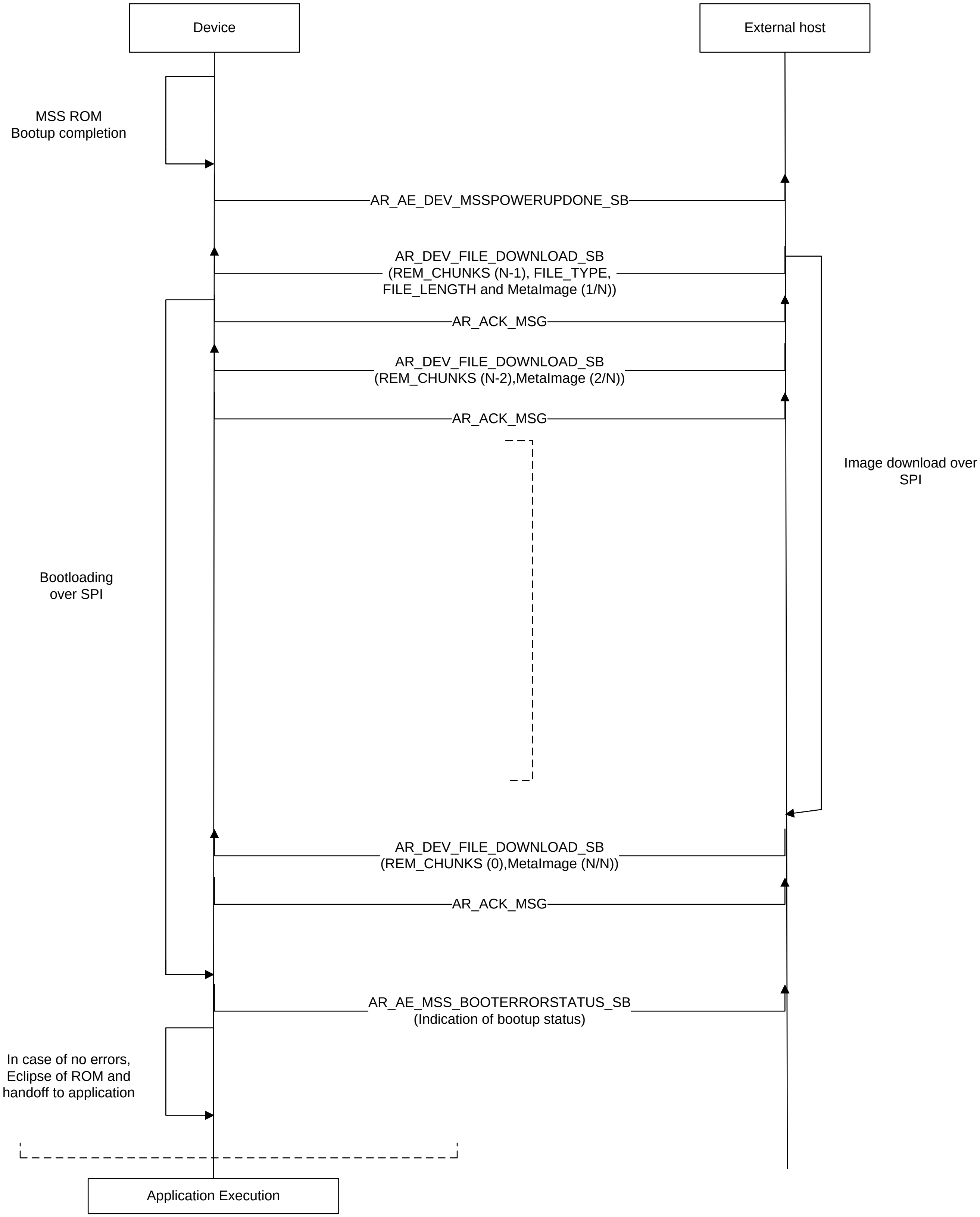

Figure 7. Bootmode – SPI

3 Programming Serial Data Flash Over UART (Bootloader Service)

The IWR1642 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 IWR1642 over the QSPI port.

SDF programming supports downloading meta images that are a combination of the following components:

- User application image for R4F (master subsystem)
- User application image for C674 (DSP subsystem)
- TI Radar Block patches

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

- MSS_UARTA of the IWR1642 device:
  - RX: Ball N4
  - TX: Ball N5
- Baud rate: 115200
- Packet size: 256 bytes

3.1 Binary File Format

The target binary file is composed of the following sections:

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

The mmWave SDK package for the IWR1642 device from TI includes the *Image Creator* utility, which constructs the complete image with the previously listed components.

3.2 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 IWR1642).
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>

3.3 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 8 the supported commands that can be issued to the IWR device during the flash programming process and the various responses from the IWR device.

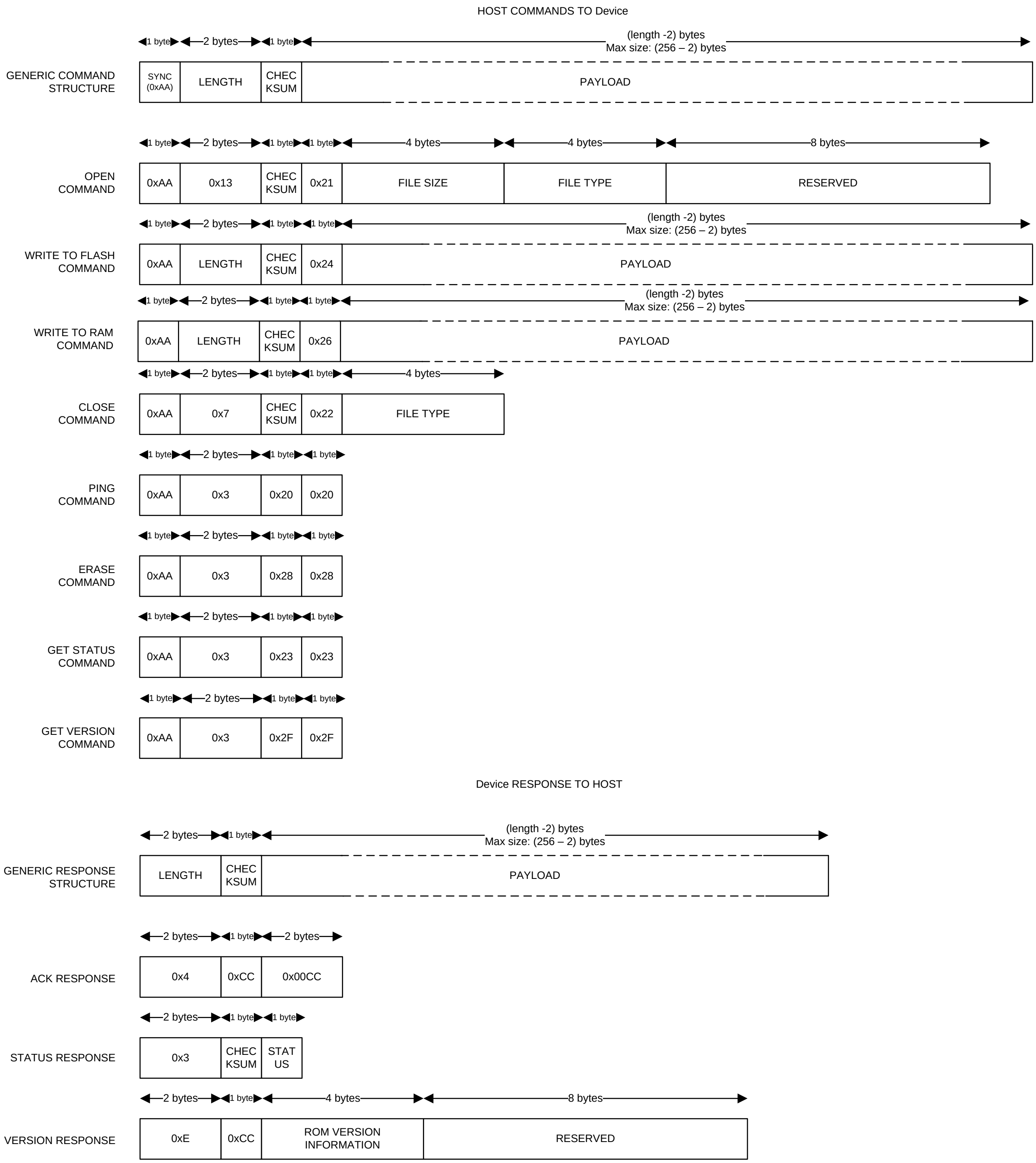

Figure 8. Host ← → IWR Device UART Communication

3.4 Flashing Sequence

Figure 9 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 9.

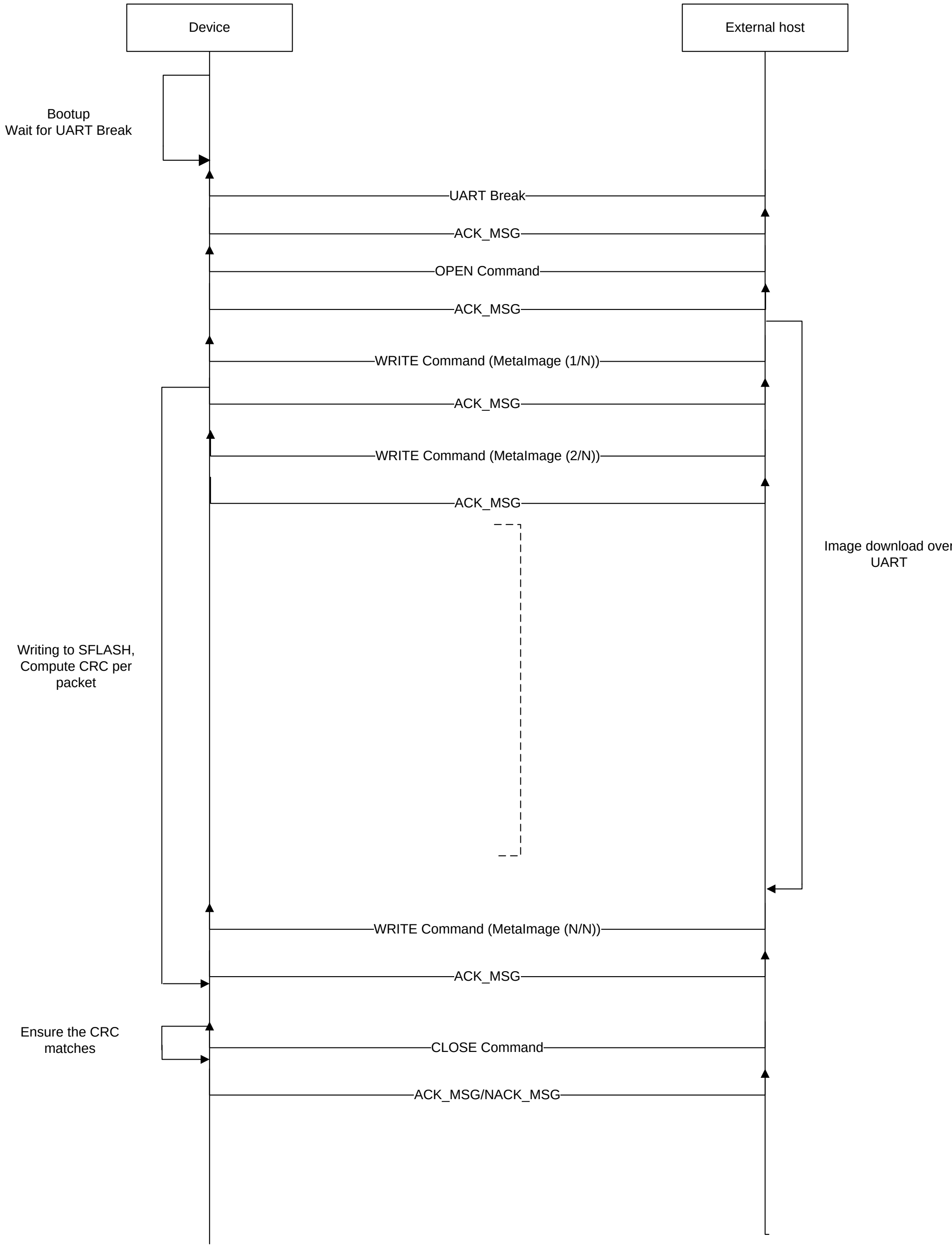

Figure 9. Flashing Sequence

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).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265