Application Note

QSPI Addressing Mode Considerations for AM62Ax Devices

Linjun Meng, Tao Han *Central FAE*

ABSTRACT

AM62Ax processors support booting from QSPI NOR Flash through an on-chip Boot ROM that uses a fixed 3-byte addressing mode during start-up. Some high-capacity Flash devices support both 3-byte and 4-byte addressing modes and can retain their addressing configuration across reset events when Flash power remains applied. Under certain conditions, this behavior can lead to a compatibility mismatch between the Boot ROM and Flash device, resulting in certificate authentication failure and boot interruption.

This application note describes the failure mechanism, presents a practical debug methodology based on Boot ROM logs, CCS memory inspection, and QSPI bus analysis, and discusses mitigation strategies for affected systems. The information is intended for hardware and software engineers developing AM62Ax-based industrial, automotive, and embedded systems that boot from QSPI NOR Flash. By following the guidance provided in this document, designers can more efficiently isolate start-up failures and improve boot reliability when using Flash devices that support 4-byte addressing mode.

The information is intended for hardware and software engineers developing AM62Ax-based industrial and embedded systems that boot from QSPI NOR Flash and can be used to improve start-up reliability and accelerate root-cause isolation.

Table of Contents

List of Figures

List of Tables

Trademarks

Jacinto™ and Sitara™ are trademarks of Texas Instruments.

All trademarks are the property of their respective owners.

1 Introduction

1.1 Purpose

QSPI NOR Flash is widely used as a boot device in AM62Ax-based systems because QSPI NOR Flash provides a compact footprint, low pin count, and straightforward boot architecture. During system development, designers must understand how Flash addressing-mode behavior interacts with Boot ROM operation under different reset conditions.

Certain Flash devices can retain their configuration when power remains applied during processor reset. If the retained Flash state differs from Boot ROM expectations, start-up compatibility issues can occur before software execution begins. Because these failures occur during the Boot ROM stage, traditional software-debug methods provide limited visibility into the root cause.

This application note describes these considerations and presents a practical methodology for diagnosing and mitigating such issues.

1.2 Scope

This document applies to:

- AM62Ax processors booting from QSPI NOR Flash
- Flash devices supporting both 3-byte and 4-byte addressing modes
- Systems implementing MCU_PORz or warm-reset mechanisms
- Flash devices capable of retaining configuration state across reset events

2 Detailed Description

2.1 AM62Ax QSPI Boot Overview

2.1.1 Boot Preconditions

Before Boot ROM can successfully access QSPI Flash, the following conditions must be satisfied.

Table 2-1. Boot Preconditions Check List

Item	Requirement
Reference Clock	Valid MCU_OSC0_XI input
Boot Configuration	BOOTMODE configured for QSPI Boot
QSPI	Functional communication with Flash device

Verification of these conditions must be the first step during start-up debugging.

2.1.2 Boot ROM Read Sequence

After reset release, the AM62Ax Boot ROM performs the following sequence:

1. Read Flash JEDEC ID using command 0x9F.
2. Issue Quad Output Fast Read command 0x6B.
3. Read tiboot3.bin starting from address 0x000000.
4. Validate the certificate and image header.
5. Transfer execution to the next boot stage.

The Boot ROM uses fixed communication parameters:

- Read command: 0x6B
- Transfer mode: 1S-1S-4S
- Address phase: 3-byte
- Dummy cycles: 8
- Clock frequency: 50MHz

These parameters are hard-coded in the Boot ROM implementation and cannot be modified by software.

2.2 Flash Addressing Architecture

2.2.1 3-Byte Addressing Mode

In 3-byte addressing mode, the Flash device uses a three-byte address phase and supports access to the first 16MB of memory space.

Most QSPI NOR Flash devices enter 3-byte addressing mode automatically following power-up or hardware reset.

2.2.2 4-Byte Addressing Mode

High-capacity Flash devices typically support 4-byte addressing mode to extend addressable memory beyond the first 16MB.

This mode is typically entered through software commands and remains active until the mode is explicitly exited or reset.

2.2.3 State Retention Behavior

The behavior of configuration registers varies among Flash vendors and device families.

For some devices, configuration state is retained while power remains applied.

For Flash devices without an independent RESET# pin, the addressing mode can remain unchanged following MCU_PORz reset if Flash power is not removed.

2.3 Boot Compatibility Considerations

2.3.1 Boot ROM Expectations

During start-up, the AM62Ax Boot ROM always assumes that the Flash device is operating in 3-byte addressing mode.

The Boot ROM issues command 0x6B followed by a fixed 3-byte address phase and eight dummy cycles. The Boot ROM does not provide any mechanism to detect the current addressing mode of the Flash device.

2.3.2 Addressing-Mode Mismatch Mechanism

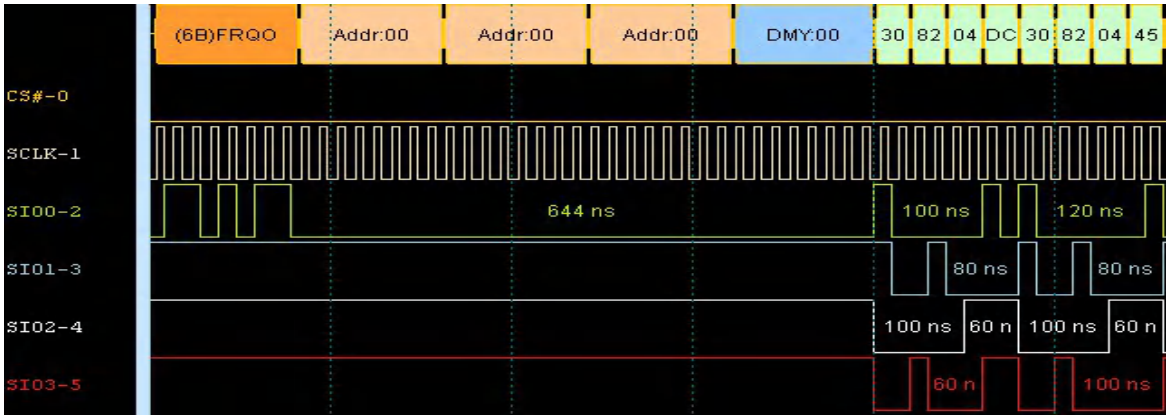

Figure 2-1. QSPI Read Timing in 3-Byte Addressing Mode

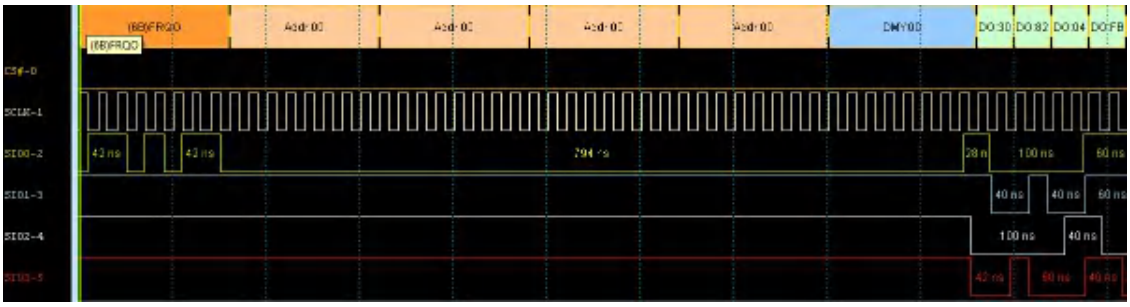

Figure 2-2. QSPI Read Timing in 4-Byte Addressing Mode

When the Flash remains configured in 4-byte addressing mode, the transaction generated by the Boot ROM is interpreted differently from the intended protocol.

The Boot ROM transmits:

Command + 3-Byte Address + Dummy Cycles + Data

However, a Flash device configured for 4-byte addressing mode interprets the same transaction as:

The Flash interprets the final byte of the Boot ROM dummy period as the fourth address byte.

As a result, part of the Boot ROM dummy period is interpreted as address information, causing Flash data output to shift relative to the Boot ROM sampling point.

2.3.3 Start-Up Impact

The addressing-mode mismatch introduces a data displacement in the image loaded by the Boot ROM.

In the investigated system, image content became shifted by one byte. As a result, the certificate header contained in tiboot3.bin is no longer located at the expected offset, causing certificate authentication to fail even though the original image and certificate remained valid.

The resulting failure sequence is shown in Figure 2-3.

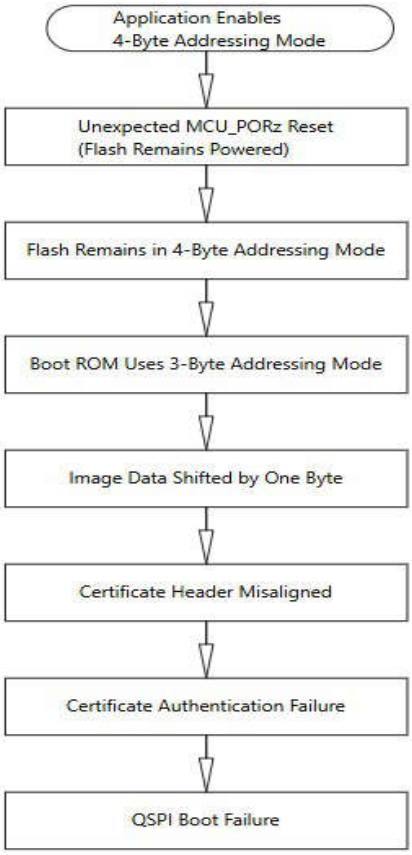

Figure 2-3. Impact of Flash Addressing-Mode Retention on the Boot Process

2.4 Example Start-Up Investigation

2.4.1 External Failure Symptoms

The investigated platform exhibited the following behavior shown in Table 2-2.

Table 2-2. Reset Behavior Comparison

Reset Condition	Result
Cold Power-On	Successful Boot
Linux Reboot	Successful Boot
MCU_PORz Reset	Startup Failure

When UART Boot was configured as the backup boot source, the device automatically switched to UART Boot after QSPI Boot failure.

This behavior provided an external indication that the failure occurred during the Boot ROM stage.

2.4.2 Boot ROM Log Analysis

Figure 2-4 shows a Boot ROM decoded failure log.

```
Warning Log:
Log Max Index Size: 21
Log current entry: 4
Log buffer: 0x43c3e480
0: main.c:1397 ecode: 0x2010018, ts: 445
"Main_derPanic()!!!"
1: main.c:1752 ecode: 0xffffffff, ts: 450
"derOneParse() failed"
2: main.c:1397 ecode: 0x2010018, ts: 679
"Main_derPanic()!!!"
3: main.c:1752 ecode: 0xffffffff, ts: 683
"derOneParse() failed"
```

Figure 2-4. Boot ROM Authentication Failure Log

Using the TI ROM Boot log dump utility, the Boot ROM failure log was captured and decoded.

The decoded log reported a certificate authentication failure.

Although this message initially suggested an image-signing or certificate-generation issue, subsequent verification confirmed that the original tiboot3.bin image and certificate were valid. Therefore, the authentication failure was treated as a symptom rather than the root cause.

2.4.3 Memory Verification Using CCS

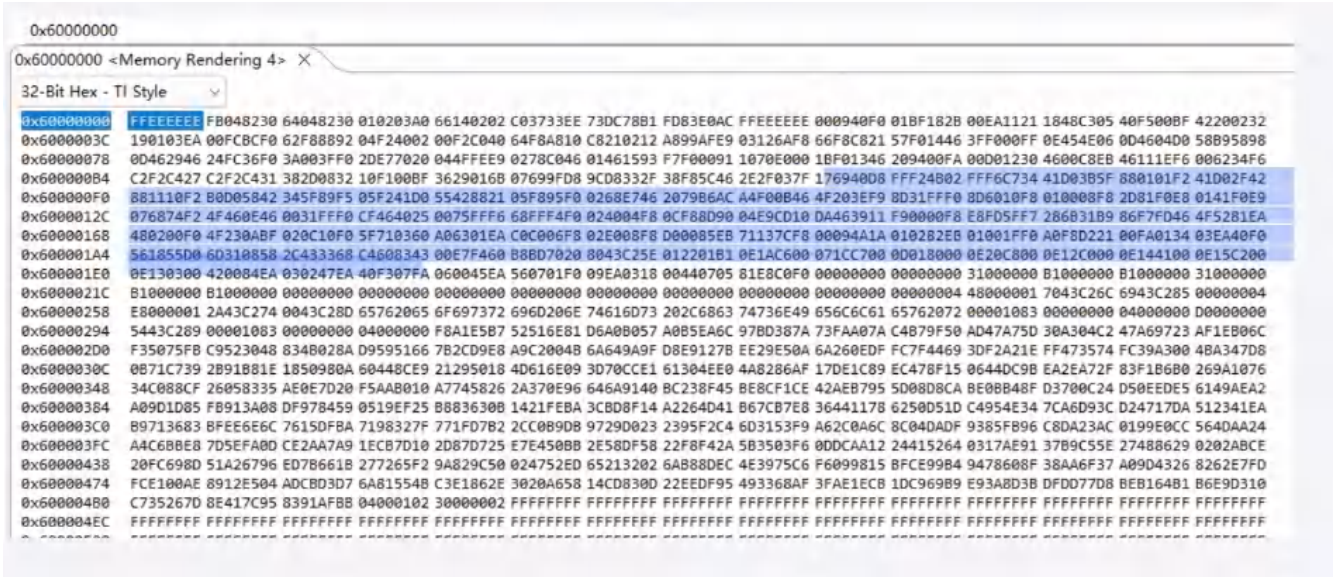

Figure 2-5. Boot ROM-Loaded Memory Content

The failing system was connected to CCS and memory contents loaded by the Boot ROM were inspected.

The expected certificate identifier was not present at the expected offset. The first bytes of memory differed from the corresponding bytes in the original tiboot3.bin image. Comparison against a successful boot case showed that the first byte of the certificate header was displaced by one byte.

This observation demonstrated that image corruption occurred before certificate validation began.

2.4.4 Image Comparison

A comparison was performed between:

- The original tiboot3.bin image
- The memory contents loaded by the Boot ROM

The comparison revealed a consistent one-byte displacement across the image.

Because the certificate header must appear at a fixed offset, the displacement directly resulted in authentication failure.

2.4.5 QSPI Bus Analysis

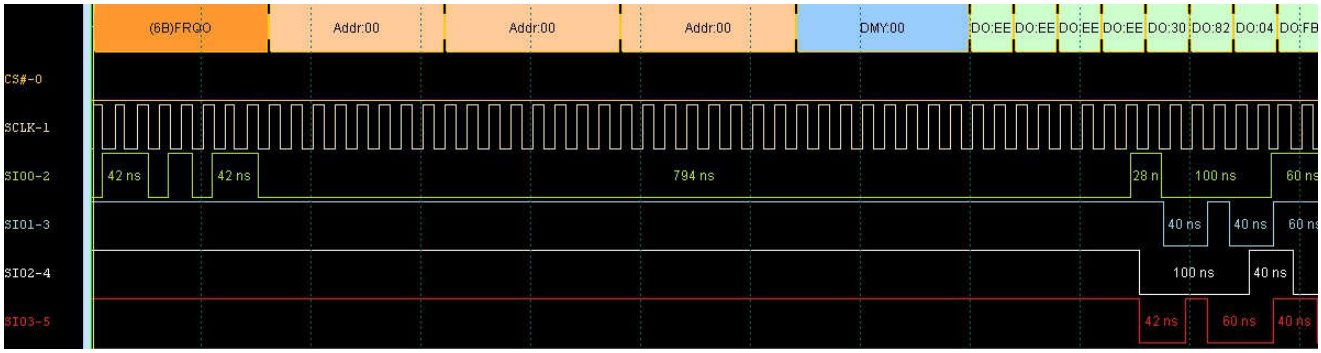

Figure 2-6. QSPI Bus Capture During Start-Up

Logic-analyzer captures confirmed that:

- The Boot ROM transmitted command 0x6B using a 3-byte address phase.
- The Flash device remained in 4-byte addressing mode.
- Flash data output began one byte later than expected.
- The observed displacement matched the offset identified during CCS memory inspection.

This correlation provided direct evidence that the authentication failure originated from Flash-read corruption rather than certificate-generation or image-signing errors.

2.4.6 Root Cause Confirmation

The combined evidence from:

- Boot ROM log analysis
- CCS memory inspection
- Image comparison
- QSPI bus analysis

confirmed that the start-up failure originated from an addressing-mode mismatch between the Boot ROM and the Flash device.

The Boot ROM always accesses Flash using a 3-byte address phase, while the Flash device remained configured for 4-byte addressing mode after MCU_PORz reset. The mismatch introduced a one-byte displacement in the image stream and ultimately caused certificate authentication failure.

2.5 Debug Methodology

The following methodology is recommended whenever a start-up failure occurs during the Boot ROM stage.

1. Step 1: Capture Boot ROM Logs
   - Determine whether the device enters a backup boot mode and collect Boot ROM logs using the appropriate TI utility.
2. Step 2: Verify Authentication Errors

- Determine whether the reported authentication failure is caused by a certificate issue or corrupted image data.

3. Step 3: Inspect Boot ROM-Loaded Memory
   - Use CCS to inspect memory populated by the Boot ROM.
4. Step 4: Compare Against the Original tiboot3.bin
   - Verify that the Boot ROM-loaded image exactly matches the programmed image.
5. Step 5: Analyze QSPI Bus Timing
   - Capture QSPI transactions using a logic analyzer and verify protocol timing.
6. Step 6: Verify Flash Addressing Configuration
   - Confirm the addressing mode retained by the Flash device after reset.

2.6 Mitigation and Design Recommendations

2.6.1 Preferred Solution: Flash Reset Synchronization

The preferred solution is to verify that the Flash device is reset whenever the processor is reset.

This makes sure that the Flash returns to 3-byte addressing mode before Boot ROM execution begins and provides the highest level of start-up robustness.

2.6.2 Alternative Solution: Driver-Level Mitigation

In the investigated platform, a software mitigation was implemented by standardizing Flash read operations using command 0x6C together with a 4-byte address phase.

Unlike Boot ROM command 0x6B, command 0x6C can operate correctly regardless of whether the Flash device is currently in 3-byte or 4-byte addressing mode, provided the target address remains within the lower Flash address space.

This approach eliminates dependency on the current Flash addressing state during normal software operation. No start-up failures were observed after implementing this modification during repeated MCU_PORz reset testing.

2.6.3 Applicability of the 0x6C Mitigation

The 0x6C-based mitigation is effective when the application does not require erase or program operations beyond the 16-MB address boundary.

Typical examples include:

- Boot-image storage
- Firmware storage
- Read-only configuration storage

In such systems, Flash accesses can be standardized to command 0x6C with a 4-byte address phase, eliminating addressing-mode dependency.

2.6.4 Design Constraints for Large-Capacity Flash Applications

Applications that perform erase or program operations beyond the 16MB boundary must still enter 4-byte addressing mode to access higher memory regions.

Following such operations, an unexpected reset can leave the Flash device in 4-byte addressing mode while the Boot ROM continues to assume 3-byte addressing mode.

For Flash devices without an independent RESET# pin, this condition can recreate the start-up compatibility issue described in this application note. Therefore:

- Hardware reset synchronization remains the preferred solution.
- Flash reset behavior must be considered during device selection.
- Warm-reset validation must be included in qualification testing.
- Addressing-mode transitions must be explicitly managed in software.

3 Summary

The start-up issue investigated in this document was traced to an addressing-mode mismatch between the AM62Ax Boot ROM and the Flash device.

Although the external symptom appeared as a certificate authentication failure, Boot ROM log analysis, CCS memory inspection, image comparison, and QSPI bus verification demonstrated that the actual root cause was a one-byte displacement introduced when the Boot ROM accessed a Flash device that remained in 4-byte addressing mode.

A driver-level mitigation based on command 0x6C was verified for systems operating entirely within the lower 16-MB address space. However, applications requiring erase or program operations beyond the 16MB boundary must carefully manage transitions into and out of 4-byte addressing mode.

For Flash devices without an independent RESET# pin, synchronizing Flash reset with processor reset remains the preferred solution because this verifies a known Flash addressing state before Boot ROM execution and provides the highest level of start-up robustness.

4 References

1. Texas Instruments, AM62Ax Sitara™ Processors, datasheet.
2. Texas Instruments, AM62A Sitara™ Processors Technical Reference Manual, technical reference manual.
3. Texas Instruments, Processor SDK Linux Software Developer's Guide for AM62AX.
4. Texas Instruments, AM62Ax MCU+ SDK for AM62AX 11.01.00.16.
5. Infineon, 256 Mb/512 Mb/1 Gb SEMPER™ Flash, datasheet.
6. Dosilicon, DS25M4CB QSPI Flash Data Sheet.
7. Texas Instruments, AM62A3: AM62A3 Abnormal Bottup after POR reset, E2E™ technical support forum.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you fully indemnify TI and its representatives against any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale, TI’s General Quality Guidelines, or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

TI objects to and rejects any additional or different terms you may propose.

Last updated 10/2025