Technical Note

NXP S32R294: Non-secure boot

May 2023

iSYSTEM

A TASKING Company

This Technical Note describes how to achieve a non-secure boot on S32R294:

- when booting from QSPI flash – using the attached Python script;
- when downloading the application directly to RAM – using the attached initialization sequence.

Tool requirements:
✓ winIDEA 9.17.140 and newer (with a matching SDK)
✓ BlueBox iC5700
✓ MPC5xxx/SPC5 Aurora Active Probe

When needed to boot the device in non-secure mode (e.g., on S32R294 trace is enabled by boot ROM when it ends booting from QSPI in non-secure mode), a specific sequence must be performed. To boot the SoC from QSPI, a valid QSPI image must be present in the QSPI flash (not just application code, but a valid IVT image).
After that, the attached Python script is run, which temporarily selects non-secure mode and forces boot. Alternatively, eFuses can be programmed as a permanent solution to select non-secure boot, e.g., when the target is expected to run standalone.
In early development, it might be more convenient to load the application directly to RAM, rather than building an IVT image and programming the QSPI flash every time there is a change in the application. However, this is not treated as a "valid boot" process by the CPU, so the SEC_BOOT register is not written to, and consequently, the trace will not be enabled. A simple initialization sequence can be executed from winIDEA to overcome this issue.

Python script and Initialization sequence script examples

The scripts should be used as a reference and changed according to the required operation (READ or WRITE).
Download and unzip script examples via links below:

S32R294_Non_Secure_Boot.7z	
SEC_BOOT.7z	

Sequence for non-secure boot

From QSPI Flash

1. Copy the script to your workspace folder.

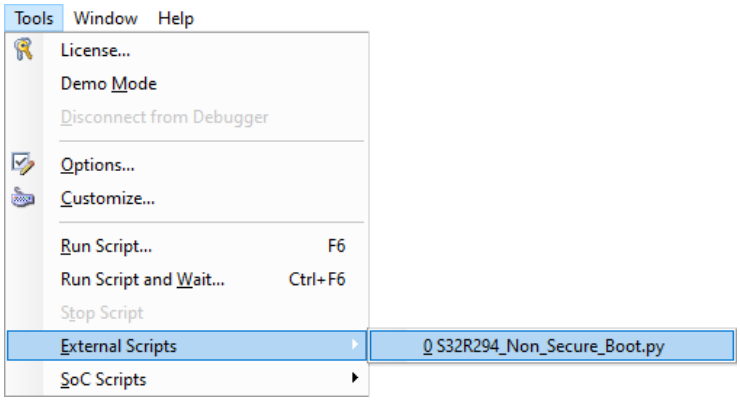
2. Open your workspace in winIDEA.

3. Disable *Latch Target RESET* option in *Hardware / CPU Options / Reset*.

4. Start a debug session (e.g., using *Reset* in winIDEA).

5. Program the QSPI flash with a valid IVT image (not necessary if the image is already programmed).

6. Run the Python script (*Tools / External scripts / S32R294_non_secure_boot.py*).



Explanation

The Python script selects the non-secure boot mode and sets a breakpoint, which covers the complete RAM area. After that, it resets core via MC_ME register.
Then, the CPU (in executing the boot code) does the following:

- After reset, the QSPI image is read.
- Then the application is copied to RAM.
- The PC is preset to the entry point and the CPU is put into running.
- CPU stops because of the breakpoint.

The Python script has a delay of 500 ms so that the CPU can execute the sequence above. After the delay, the breakpoint is removed and CPU context is invalidated (so that the right PC value is displayed).
When the target is intended to run standalone, i.e., when the IVT image is valid and the application runs correctly, the user can opt to program the eFuses to enable the SEC_BOOT. This means that at the boot time, eFuse will be read and written to the SEC_BOOT register, and no intervention from the debugger is necessary.

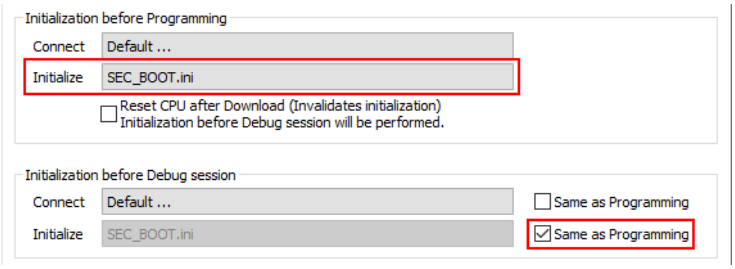
Downloading directly to RAM

1. Copy the *SEC_BOOT.ini* file to your workspace folder.

2. Open your workspace in winIDEA.

3. Open *Hardware / CPU Options / Reset* and add the *SEC_BOOT.ini* script in *Initialization before Programming / Initialize*.

4. Check the *Same as Programming* check box in *Initialization before Debug session / Initialize*.



Explanation

When the boot ROM finishes non-secure booting, it writes to the SEC_BOOT register. When the application is loaded directly to RAM, the boot ROM does not execute, and therefore the SEC_BOOT register is not written to. By specifying the *SEC_BOOT.ini* sequence in winIDEA, we ensure that the SEC_BOOT register is written to by the debugger, which is then treated by the CPU as if the boot ROM has finished booting.

Additional resources

Script, which can be used for eFuse programming, can be found in the Technical Note S32R294 eFuse Programming.

To select Secure boot mode on S32R294, eFuse 0x240 must be programmed with value 0xC0.