AURIX™ TC3xx Device Locking

Dos and Don’ts

Common Root Causes
Which Lead to a Locked Device

DON'TS

- UCB content corrupted with multiple ECC errors
  - Due to voltage drop during programming or erasing phase
- UCB with errored or erased confirmation code
  - Due to voltage drop or wrong UCB handling
- HSMBOOTEN set without a valid HSM boot table programmed
  - ARM core needs valid boot code to be able to jump to the user code
- FLASH HSM sectors wrongly erased with HSM enabled
  - Due to voltage drop during programming or erasing phase
- Wrong UCB handling with incorrect debugger scripts
- SWAP enabled with HSM enabled but without valid boot code in the alternate bank
- UCB_HSMCOTP with a wrong PROCONHSMCBS set
  - where HSM should find its boot code

Common Root Causes
Which Lead to a Locked Device

DON'TS

- Following points are the root cause to lock the debugger interface without intention with the wrong assumption that the device is bricked:
  - UCB_BMHD with valid content not found and HSM enabled with valid boot code
    - The device behavior changes when HSM is enabled and BMHD are not programmed. Situation is recoverable but with specific debugger scripts.
  - Read Protection applied to PFLASH/DFLASH
    - User needs to unlock the debugger interface via internal routine to clear the the OSTATE.IF_LCK bit writing the CBS_OEC register.
  - UCB_DBG with only PROCONDBG.OCDSDIS bit set
    - This can disable the OCDS with the effect to not be able to set breakpoints and halt the core anymore, till the OSTATE.IF_LCK bit is cleared.

Lock Condition
Avoidance

DOS

In order to avoid a lock condition, please ensure that:

1. Write a valid BMHD in your application for the TriCore side (This can be done with a lauterbach script if this is not included in your TriCore application code).
2. Before enabling the HSM, program a valid HSM User code and HSM boot code.
3. Provide the correct address where HSM has to search for its boot code (you can use the lauterbach script to program the PROCONHSMCBS register).
4. Inside the HSM boot code, define the address where the HSM will execute the HSM user code.
5. Separate, via locator file, PFLASH sectors dedicated for HSM code from the one used by HOST. This lead to have an HOST start address from 0xA00A0000.
6. Reserve HSM Flash sector 0 to write a valid HSM boot code and start HSM code from sector 1. Select BootSel0 accordingly.

Useful Appnotes

Please follow these steps in order to get started:

1. **Register** for myInfineon
2. For AURIX documentation please send email with your Myinfineon credentials to **AURIX@infineon.com**
3. You will receive a confirmation which explains how to use your new access

AURIX™ TC3xx Avoiding device lockups

https://myicp.infineon.com/sites/microcontrollers-aurix_customer_doc/Lists/defaultdoclib/AURIX%20TC3xx/10%20AURIX%20TC3xx%20HSM/06%20Application%20Notes/AP32555-Tutorial_AURIX_TC3xx_Avoiding_device_lockups_v1.0.1.pdf


Infineon