nRF5340 Production Programming

Application Note

4406_642 / 2023-02-14

NORDIC® SEMICONDUCTOR

Contents

Revision history

Date	Description
2022-09-14	• Corrected Register addresses for ERASEPROTECT and SECUREAPPROTECT in Enabling device protection on page 11 • Editorial changes
2021-01-29	First release

1 Introduction

This document provides information on writing software to nRF5340 devices and is intended for developers of flash programming tools.

It serves as a starting point for nRF5340 device support in production tools and accelerates the engineering process of supporting nRF5340 devices. This document describes a robust way to program devices. You might not need to follow every step in some cases (for example, if the device has never been programmed before and its flash is completely erased, or if the device is unprotected).

2 Programming flow

The diagram shows the flow of production programming under normal circumstances.

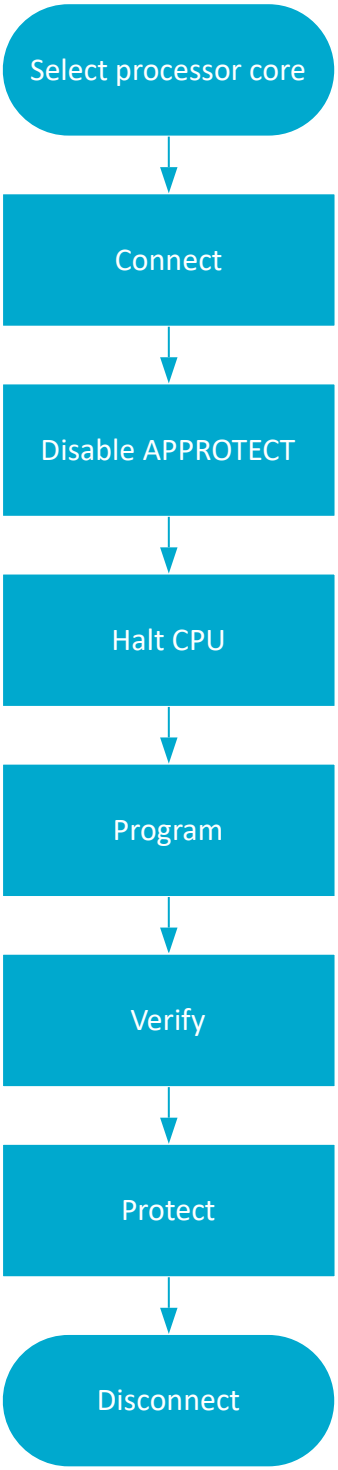

Figure 1: Normal programming flow

3 Selecting the processor core

The nRF5340 has two processors cores, the application core and the network core.

When communicating with the device using the *Serial Wire Debug Port (SW-DP)*, the application core at AHB-AP=0 is selected by default. The network core can be found at AHB-AP=1. To select the target processor, set AHB-AP to the corresponding value.

4 Disabling APPROTECT

If the device has access port protection enabled and is not erase protected, you can disable access port protection with a *Control Access Port (CTRL-AP)* erase all operation.

1. Write `1` to CTRLAP.ERASEALL.
2. Wait until the value of CTRLAP.ERASEALLSTATUS becomes `1`.

Note: nRF5340 devices have two CTRL-APs, one for each core.

5 Halting the CPU

Use the standard *Serial Wire Debug (SWD)* Arm® CoreSight™ *Debug Access Port (DAP)* protocol to issue a halt command to the chip.

An application running on the device that was previously programmed may use the *Watchdog timer (WDT)*. The default configuration of the WDT will pause it, if the CPU is halted.

6 Connecting

Use the standard *SWD* Arm CoreSight *DAP* protocol to enter debug interface mode.

Before the external debugger can access the CPU, it must first request the device to power up and make sure that the appropriate power domains are powered up. This is handled using the built-in CxxxPWRUPREQ and CxxxPWRUPACK feature found in the DAP. As long as the debugger is requesting the debug domain or the complete system to power up, the device stays in debug interface mode.

Before connecting to the network core, check if it is in Force-OFF mode and not held in reset. To check if the network core is powered up, do an AHB read of AHB-AP 0 by targeting `RESET.NETWORK.FORCEOFF (0x50005614)` in the application core. If the readout is `0`, the network core is not in Force-OFF mode. If the readout is `1`, write `0` to it to exit Force-OFF mode.

7 Programming

When writing is enabled, the non-volatile memory is written by writing a word to a word-aligned address in the code or *User Information Configuration Registers (UICR)*. Only word-aligned writes are allowed. Byte or half-word-aligned writes result in a hard fault.

7.1 Writing data - SECUREAPPROTECT disabled

Use the standard *SWD* Arm CoreSight *DAP* protocol to write data into flash.

The *Non-volatile Memory Controller (NVMC)* peripheral has two different base addresses:

- Application core – `0x50039000`
- Network core – `0x41080000`

1. Set the CONFIG register of the NVMC to `WEN.Wen` by writing `0x00000001` to the following addresses:
   - Application core - 0x50039504
   - Network core - 41080504

   This enables writing to the non-volatile memory.
2. Read the READY register (0x50039400) of the NVMC until the value is `0x00000001`. Read the following addresses:
   - Application core - 0x50039400
   - Network core - 0x41080400

   When this value is read, the NVMC is ready and not currently performing any operations.
3. Write the data to the desired, word-aligned address.
4. Read the READY register of the NVMC until the value is `0x00000001` before continuing to ensure the write operation has completed. Read the following addresses:
   - Application core - 0x50039400
   - Network core - 0x41080400
5. Continue writing and then reading the READY register as necessary.
6. Set the CONFIG register of the NVMC to `WEN.Ren` by writing `0x00000000` to the following addresses:
   - Application core - 0x50039504
   - Network core - 41080504

   This configures the non-volatile memory as read-only.

The ranges of writeable addresses are:

- *UICR* addresses (located in addresses 0x00FF8000 through 0xFF8FFC for the application core, 0x01FF8000 through 0x01FF8FFC for the network core)
- All program flash (located in addresses 0x00000000 through ((INFO.CODESIZE * INFO.CODEPAGESIZE) – 0x4)

You can write to flash using different methods. For nRF5340, a good flash algorithm should take around 11 seconds to write the entire flash to the application core, and around 3 seconds for the network core. If you cannot achieve this time, contact Nordic Semiconductor for assistance.

7.2 Verifying flash content

To verify the contents of flash after programming, use the standard *SWD* Arm CoreSight *DAP* protocol to read every address written and compare to expected values.

7.3 Enabling device protection

There are several ways to protect nRF5340 devices. *Access Port Protection (APPROTECT)* secures the access port, *Erase Protection (ERASEPROTECT)* stops the device from being erased, and *Secure Access Port Protection (SECUREAPPROTECT)* stops unauthorized access to the secure domain.

Note: If the device has activated both APPROTECT and ERASEPROTECT, it cannot be recovered without a proper software solution. See Checking protection status on page 13 for more information.

APPROTECT

APPROTECT blocks debugger read/write access to all CPU registers and memory mapped addresses.

To check if APPROTECT is already enabled, read the UICR.APPROTECT(0x00FF8000) register. If this register has a value other than `0x50FA50FA`, it is protected. If the register shows the device as unprotected, write any value other than `0x50FA50FA` to it. The protection activates after a reset. If you are activating ERASEPROTECT or SECUREAPPROTECT, wait to reset until all the protections are set.

ERASEPROTECT

ERASEPROTECT blocks NVMC ERASEALL and CTRL-AP.ERASEALL functionality.

To check if ERASEPROTECT is already enabled, read the UICR.ERASEPROTECT(0x00FF8020) register. If this register has a value other than `0xFFFFFFFF`, it is protected. If the register shows the device as unprotected, write any value other than `0xFFFFFFFF` to it. The protection activates after a reset. If you are activating APPROTECT or SECUREAPPROTECT, wait to reset until all the protections are set.

SECUREAPPROTECT

SECUREAPPROTECT blocks debugger read/write access to all secure CPU registers and secure memory mapped addresses.

To check if SECUREAPPROTECT is already enabled, read the UICR.SECUREAPPROTECT(0x00FF801C) register. If this register has a value other than `0x50FA50FA`, it is protected. If the register shows the device as unprotected, write any value other than `0x50FA50FA` to it. The protection activates after a reset. If you are activating *APPROTECT* or ERASEPROTECT, wait to reset until all the protections are set.

8 Disconnecting

Use the standard *SWD* Arm CoreSight *DAP* protocol to exit the debug interface mode.

This is handled using the built-in CxxxPWRUPREQ and CxxxPWRUPACK features found in the Arm CoreSight DAP. When the debugger stops requesting the debug domain or the complete system to be powered up, the device exits the debug interface mode.

We recommend a hard reset of the device after programming by doing a pin reset or a power cycle.

9 Troubleshooting

nRF5340 devices can be reprogrammed by erasing pages. Reprogramming is also dependent on the current nRF5340 security setting.

9.1 Checking protection status

nRF5340 devices have access port protection enabled by default, but they can also have secure access port protection and erase protection.

For more information on access port protection, see Enabling device protection on page 11.

To check if your device is protected, read the following registers in the AHB-AP:

- AHP-AP Control/Status Word (CSW) register – use this register to read the *APPROTECT* status. This register is defined in the Arm® CoreSight SoC-400 Technical Reference Manual. Use the following fields to check the access port protection status:
  - DgbStatus field (bit 6 in AHB-AP.CSW) – indicates if AHB transfers are permitted. If the value of AHB-AP.CSW->DbgStatus is `1`, then AHB transfers are allowed and the device does not have APPROTECT.
  - SPIStatus field (bit 23 in AHB-AP.CSW) - indicates if secure protection is enabled. If the value of AHB-AP.CSW->SPIStatus is `1`, *SECUREAPPROTECT* is not set and both secure and non-secure transfers are allowed.
- CTRL-AP.ERASEPROTECT.STATUS – If access port protection is not enabled, use this register to check if *ERASEPROTECT* is set. If both APPROTECT and ERASEPROTECT are set, the device cannot be unlocked unless programmed software changes the settings.

9.1.1 APPROTECT and ERASEPROTECT are enabled

If both *APPROTECT* and *ERASEPROTECT* are enabled, access port 0 and the ERASEALL functionality are unavailable.

To unlock the device, it must have compatible firmware that provides a 32-bit non-zero KEY value to ERASEPROTECT.DISABLE. When both the debugger and firmware provide the same 32-bit non-zero KEY value to ERASEPROTECT.DISABLE, the device does a *CTRL-AP* erase all operation. The access port is re-enabled on the next reset once the erase sequence is done.

9.1.2 Only APPROTECT is enabled

If *APPROTECT* is enabled on the device, access port `0` is unavailable.

The only way to reopen or unlock the device is to issue an ERASEALL command through the *CTRL-AP* access port and then issue a pin reset. This will erase the entire code flash, the *UICR* area of the device, and the entire RAM. This method of erasing is slower than performing an *NVMC* erase all since it also must erase all RAM, but if APPROTECT is enabled, it is the only way to unlock the device.

9.1.2.1 Erasing all through CTRL-AP

Use the standard *SWD* Arm CoreSight *DAP* protocol to erase all while *CTRL-AP* is still selected by the debug port.

1. Write `0x00000001` to the ERASEALL register (0x004) of CTRL-AP.

   This will start the ERASEALL operation which erases all flash and RAM on the device.
2. Read the ERASEALLSTATUS register (0x008) of the CTRL-AP until the value read is `0x00` or wait 15 seconds after the ERASEALL write has expired.

3. Issue a pin reset.

9.1.2.2 Halting the CPU

Use the standard *SWD* Arm CoreSight *DAP* protocol to issue a halt command to the chip.

An application running on the device that was previously programmed may use the *WDT*. The default configuration of the WDT will pause it, if the CPU is halted.

9.1.2.3 Reloading the watchdog timer

If the *WDT* is configured not to stop on a halt command, it must be reloaded periodically to prevent it from resetting the domain.

1. Read the WDT.RUNSTATUS register (0x50018400) to check if the WDT is running.

   If the least significant bit is `1`, the WDT is running.
2. Read the **HALT** field of the WDT.CONFIG (0x5001850C) register to check if the WDT halts with the CPU.

   If the fourth least significant bit is `1`, the WDT does not halt.
3. Reload the WDT by doing the following:
   a) Identify an enabled reload request in WDT.RREN (0x50018508).

      If value of the bit `n` in WDT.RREN is `1`, the reload request `n` in WDT.RR[n] (0x50018600 + (n × 0x4)) can be used to reload the watchdog counter to the value written in WDT.CRV (0x50018504).
   b) Reload the watchdog timer by writing `0x6E524635` to the WDT.RR[n] register.

9.1.2.4 Reading FICR

*Factory Information Configuration Registers (FICR)*s are pre-programmed in the factory and cannot be erased by the user. These registers contain chip-specific information and configuration.

Using the standard *SWD* Arm CoreSight *DAP* protocol:

1. Read the INFO.CODEPAGESIZE register (0x00FF0220) of the FICR.

   The value of this register contains the code memory page size in hexadecimal format, so `0x00001000` stored in this register corresponds to a page size of 4096 bytes.
2. Read the INFO.CODESIZE register (0x00FF0224) of the FICR.

   The value of this register contains the number of pages in code memory in hexadecimal format, so `0x0000100` stored in this register corresponds to 256 total pages in flash memory.

> **Note:** Total flash memory (in bytes) = INFO.CODEPAGESIZE * INFO.CODESIZE. This information is used later to determine the valid range of addresses to program.

9.1.3 APPROTECT is disabled

The *UICR*s have not been previously configured to enable access port protection.

If the device is in secure protection, you cannot read *FICR*s or disable the *System Protection Unit (SPU)*. To disable *SECUREAPPROTECT* , see Erasing all through CTRL-AP on page 13.

In some cases, you may assume that the entire flash has already been erased. If the flash is already erased and the device has never been programmed before, go to Programming on page 10.

9.1.3.1 Reading FICR

*FICR*s are pre-programmed in the factory and cannot be erased by the user. These registers contain chip-specific information and configuration.

Using the standard *SWD* Arm CoreSight *DAP* protocol:

1. Read the INFO.CODEPAGESIZE register (0x00FF0220) of the FICR.

The value of this register contains the code memory page size in hexadecimal format, so `0x00001000` stored in this register corresponds to a page size of 4096 bytes.

2. Read the INFO.CODESIZE register (0x00FF0224) of the FICR.

   The value of this register contains the number of pages in code memory in hexadecimal format, so `0x0000100` stored in this register corresponds to 256 total pages in flash memory.

Note: Total flash memory (in bytes) = INFO.CODEPAGESIZE * INFO.CODESIZE. This information is used later to determine the valid range of addresses to program.

9.1.3.2 Halting the CPU

Use the standard *SWD* Arm CoreSight *DAP* protocol to issue a halt command to the chip.

An application running on the device that was previously programmed may use the *WDT*. The default configuration of the WDT will pause it, if the CPU is halted.

9.1.3.3 Reloading the watchdog timer

If the *WDT* is configured not to stop on a halt command, it must be reloaded periodically to prevent it from resetting the domain.

1. Read the WDT.RUNSTATUS register (0x50018400) to check if the WDT is running.

   If the least significant bit is `1`, the WDT is running.
2. Read the **HALT** field of the WDT.CONFIG (0x5001850C) register to check if the WDT halts with the CPU.

   If the fourth least significant bit is `1`, the WDT does not halt.
3. Reload the WDT by doing the following:
   a) Identify an enabled reload request in WDT.RREN (0x50018508).

      If value of the bit `n` in WDT.RREN is `1`, the reload request `n` in WDT.RR[n] (0x50018600 + (n × 0x4)) can be used to reload the watchdog counter to the value written in WDT.CRV (0x50018504).
   b) Reload the watchdog timer by writing `0x6E524635` to the WDT.RR[n] register.

9.1.3.4 Disabling SPU

The *SPU* protects memories from illegal erases, writes, and accesses.

An attempt to erase, write, or access a protected memory region leads to a secure fault. System protection can be turned off in the debug mode by configuring the relevant FLASHREGION[n].PERM for flash and RAMREGION[n].PERM for RAM.

To disable the SPU, the processor must be reset and stopped in the reset vector using the Flash Patch and Breakpoint unit (FPB). This disables any SPU protection and prevents any firmware from re-enabling the SPU.

See nRF5340 SPU – System protection unit for more information.

9.1.3.5 Check flash region security attribute

For every flash page, there is a security attribute that controls if that region is secure or non-secure.

Check the corresponding SPU.FLASHREGION[n].PERM (0x5003600 + (n*4)), where n is the flash page. If the 4th bit is `0`, the flash region is in non-secure, otherwise it is secure.

9.2 Erasing

The flash can be erased by either erasing page by page or erasing all pages.

An erase all operation takes the same amount of time as erasing two pages one by one. With more than 2 flash pages, erasing all is more efficient than erasing page by page. If a region of the chip has

been preprogrammed, you can erase the flash you intend to program page by page and then write those addresses with data leaving pre-programmed flash untouched. If the value of all flash addresses is `0xFFFFFFFF`, skip this procedure.

9.2.1 Erasing all

Use the standard *SWD* Arm CoreSight *DAP* protocol to erase all pages.

1. If *SECUREAPPROTECT* is enabled, erase all is disabled. To erase, see either Erasing page by page on page 16 or Erasing all through CTRL-AP on page 13 to disable SECUREAPPROTECT.
2. Set the CONFIG register of the *NVMC* to `WEN.Een` by writing `0x00000002` to the following addresses:
   - Application core - 0x50039504
   - Network core - 41080504

   This configures the non-volatile memory for erasing.
3. Read the READY register of the NVMC until the value is `0x00000001`. Read the following addresses:
   - Application core - 0x50039400
   - Network core - 0x41080400

   When this value is read, the NVMC is ready and not currently performing any operations.
4. Set ERASEALL register of the NVMC to `ERASEALL.Erase` by writing `0x00000001` to the following addresses:
   - Application core - 0x5003950C
   - Network core - 4108050C

   This erases all non-volatile memory including *UICR* registers.
5. Read the READY register of the NVMC until the value is `0x00000001` before continuing to ensure the erase all operation has completed. Read the following addresses:
   - Application core - 0x50039400
   - Network core - 0x41080400
6. Set the CONFIG register of the NVMC to `WEN.Ren` by writing `0x00000000` to following addresses:
   - Application core - 0x50039504
   - Network core - 41080504

   This configures the non-volatile memory back to read-only.

9.2.2 Erasing page by page

Use the standard *SWD* Arm CoreSight *DAP* protocol to erase page by page.

9.2.2.1 SECUREAPPROTECT disabled on network core

Use this procedure to erase page by page if *SECUREAPPROTECT* is disabled on the network core.

Before you begin, check *SPU* if the flash page is secure or non-secure (see Check flash region security attribute on page 15).

1. Set the CONFIG register of the NVMC to `WEN.Een` by writing `0x00000002` to the following addresses:
   - Application core - 0x50039504
   - Network core - 41080504

   This configures the non-volatile memory for erasing.
2. Read the READY register of the NVMC until the value is `0x00000001`. Read the following addresses:

- Application core - 0x50039400
- Network core - 0x41080400

When this value is read, the NVMC is ready and not currently performing any operations.

3. Write `0xFFFFFFFF` to the first 32-bit word in the flash page you want to be erased.
4. Read the READY register of the NVMC until the value is `0x00000001`. Read the following addresses:
   - Application core - 0x50039400
   - Network core - 0x41080400
5. Repeat steps 3 and 4 until all wanted pages are erased.
6. Set the CONFIG register of the NVMC to `WEN.Ren` by writing `0x00000000` to the following addresses:
   - Application core - 0x50039504
   - Network core - 41080504

   This configures the non-volatile memory back to read-only.

Note: The *UICR* flash page cannot be erased by using erase page. It can only be erased by an erase all operation.

9.2.2.2 SECUREAPPROTECT enabled

Use this procedure to erase page by page if *SECUREAPPROTECT* is enabled.

Note: This procedure is only valid for the application core.

1. Write `0x00000002` to the CONFIGNS register (0x40039584) of the *NVMC*.
   This configures the non-volatile memory for erasing.
2. Read the READY register (0x40039400) of the NVMC until the value is `0x00000001`.
   When this value is read, the NVMC is ready and not currently performing any operations.
3. Write `0xFFFFFFFF` to the first 32-bit word in the flash page you want to be erased.
4. Read the READY register (0x40039400) of the NVMC until the value is `0x00000001`.
5. Repeat steps 3 and 4 until all wanted pages are erased.
6. Write `0x00000000` to the CONFIGNS register (0x40039584) of the NVMC.
   This configures the non-volatile memory back to read-only.

Glossary

Access Port Protection (APPROTECT)

A register used to prevent read and write access to all CPU registers and memory-mapped addresses.

Control Access Port (CTRL-AP)

A custom access port that enables control of the device even if other access ports in the debug access port are disabled by the access port protection.

Debug Access Port (DAP)

Provides multiple master driving ports, all accessible and controlled through a single external interface port to provide system-wide debug.

Erase Protection (ERASEPROTECT)

A register used to block NVMC ERASEALL and CTRL-AP.ERASEALL functionality.

Factory Information Configuration Registers (FICR)

Pre-programmed registers that contain chip-specific information and configuration. FICRs cannot be erased by users.

Non-volatile Memory Controller (NVMC)

A controller used for writing and erasing the internal flash memory and the *UICR*.

Secure Access Port Protection (SECUREAPPROTECT)

A register used to prevent read and write access to all secure CPU registers and secure memory-mapped addresses.

System on Chip (SoC)

A microchip that integrates all the necessary electronic circuits and components of a computer or other electronic systems on a single integrated circuit.

System Protection Unit (SPU)

The central point in the system that controls access to memories, peripherals, and other resources.

Serial Wire Debug (SWD)

A standard two-wire interface for programming and debugging Arm CPUs.

Serial Wire Debug Port (SW-DP)

An interface that provides a low pin count bi-directional connection to the DAP with a reference clock signal for synchronous operation.

User Information Configuration Registers (UICR)

Non-volatile memory registers used to configure user-specific settings.

Watchdog timer (WDT)

A timer that causes a system reset if it is not poked periodically.

Recommended reading

In addition to the information in this document, you may need to consult other documents.

Nordic documentation

The following sections in the nRF5340 Product Specification contain relevant information for programming *System on Chip (SoC)*s:

- Memory – nRF5340 devices use flash-based non-volatile memory in the code flash, *UICR*, and *FICR* memory regions. This section includes information about flash page size and number.
- NVMC — Non-volatile memory controller – has detailed specifications about timing for write/erase operations.
- Debug and trace – provides access to the on-chip debug functionality. This is a standard two-pin *SWD* interface as defined by Arm.

Other documentation

See the CoreSight Components Technical Reference Manual for more information on general concepts such as Arm CoreSight or SWD.

Legal notices

By using this documentation you agree to our terms and conditions of use. Nordic Semiconductor may change these terms and conditions at any time without notice.

Liability disclaimer

Nordic Semiconductor ASA reserves the right to make changes without further notice to the product to improve reliability, function, or design. Nordic Semiconductor ASA does not assume any liability arising out of the application or use of any product or circuits described herein.

Nordic Semiconductor ASA does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. If there are any discrepancies, ambiguities or conflicts in Nordic Semiconductor's documentation, the Product Specification prevails.

Nordic Semiconductor ASA reserves the right to make corrections, enhancements, and other changes to this document without notice.

Life support applications

Nordic Semiconductor products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury.

Nordic Semiconductor ASA customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nordic Semiconductor ASA for any damages resulting from such improper use or sale.

RoHS and REACH statement

Complete hazardous substance reports, material composition reports and latest version of Nordic's REACH statement can be found on our website www.nordicsemi.com.

Trademarks

All trademarks, service marks, trade names, product names, and logos appearing in this documentation are the property of their respective owners.

Copyright notice