nRF Connect SDK v1.2.0 Release Notes

nRF Connect SDK delivers reference software and supporting libraries for developing low-power wireless applications with Nordic Semiconductor products. It includes the MCUboot and the Zephyr RTOS open source projects, which are continuously integrated and re-distributed with the SDK.

nRF Connect SDK v1.2.0 supports product development with the nRF9160 Cellular IoT device. It contains reference applications, sample source code, and libraries for developing low-power wireless applications with nRF52 and nRF53 Series devices, though support for these devices is incomplete and not recommended for production.

Highlights

- Support for the new nRF5340 device
- New Multi-Protocol Service Layer (MPSL), included in nrfxlib
- NFC supporting the new TNEP protocol
- New nRF9160 samples and libraries
- General updates and bugfixes

Release tag

The release tag for the nRF Connect SDK manifest repository (https://github.com/NordicPlayground/fw-nrfconnect-nrf) is **v1.2.0**. Check the `west.yml` file for the corresponding tags in the project repositories.

To use this release, check out the tag in the manifest repository and run `west update`. See Getting the nRF Connect SDK code for more information.

Supported modem firmware

This version of the nRF Connect SDK supports the following modem firmware for cellular IoT applications:

- mfw_nrf9160_1.1.1

Use the nRF Programmer app of nRF Connect for Desktop to update the modem firmware. See Updating the nRF9160 DK cellular modem for instructions.

Supported boards

- PCA10090 (nRF9160 DK)
- PCA20035 (Thingy:91)
- PCA10095 (nRF5340 PDK)
- PCA10056 (nRF52840 Development Kit)
- PCA10059 (nRF52840 Dongle)
- PCA10040 (nRF52 Development Kit)

Required tools

In addition to the tools mentioned in Installing the nRF Connect SDK manually, the following tool versions are required to work with the nRF Connect SDK:

Tool	Version	Download link
SEGGER J-Link	V6.60e	J-Link Software and Documentation
nRF Command Line Tools	v10.6.0	nRF Command Line Tools
nRF Connect for Desktop	v3.3.0 or later	nRF Connect for Desktop
dtc (Linux only)	v1.4.6 or later	Installing the required tools
GCC	See Installing the toolchain	GNU Arm Embedded Toolchain

As IDE, we recommend to use SEGGER Embedded Studio (Nordic Edition) version 4.42a. It is available from the following links:

- SEGGER Embedded Studio (Nordic Edition) - Windows x86
- SEGGER Embedded Studio (Nordic Edition) - Windows x64
- SEGGER Embedded Studio (Nordic Edition) - Mac OS x64
- SEGGER Embedded Studio (Nordic Edition) - Linux x86
- SEGGER Embedded Studio (Nordic Edition) - Linux x64

Change log

The following sections provide detailed lists of changes by component.

nRF9160

- Added the following samples:
  - nRF9160: Cloud Client - shows how to connect to and communicate with a cloud service using the generic Cloud API.

  - nRF9160: HTTPS Client - shows how to provision a TLS certificate and connect to an HTTPS server.
  - nRF9160: Serial LTE Modem - demonstrates sending AT commands between a host and a client device. The sample is an enhancement to the nRF9160: AT Client sample.
- Added the following libraries:
  - AWS IoT - enables applications to connect to and exchange messages with the AWS IoT message broker. The library supports TLS-secured MQTT transmissions and firmware over-the-air upgrades.
  - Modem key management - provides functions to provision security credentials to the nRF9160 modem. The library replaces the `nrf_inbuilt_key` APIs from the BSD library.
  - China Telecom ZZHC library - implements the self-registration functionality that is required to connect to the China Telecom network.
  - SUPL client - integrates the externally hosted SUPL client library. This library implements A-GPS data downloading from a SUPL server.

Updated samples and applications

- nRF9160: Asset Tracker:
  - Added functionality to configure high/low thresholds for sensor data, so that only data below/above the threshold is sent to the cloud.
  - Modified the command format to match the format that is used by nRF Cloud.
  - Implemented support for receiving modem AT commands from the cloud and returning the modem's response.
  - Added functionality to configure the interval at which sensor data is sent to the cloud. This makes it possible to change the poll/send interval for environmental and light sensors from the terminal card in nRF Cloud.
  - Replaced `printk` calls with calls to the Logging subsystem.
  - Added a separate workqueue for the application, instead of using the system workqueue.
- nRF9160: GPS sockets and SUPL client library:
  - Added support for the new SUPL client library, if enabled.

Updated libraries

- Download client:
  - Added the `CONFIG_DOWNLOAD_CLIENT_MAX_TLS_FRAGMENT_SIZE` option that allows to configure fragment sizes for TLS connections and non-TLS connections independently.
  - Added support for using non-default ports.
- Secure Partition Manager (SPM):
  - Updated the security attribution to configure the peripherals NRF_REGULATORS and NRF_WDT as Non-secure.
  - Added the RTC0 peripheral (as Non-Secure).

  - Fixed a bug where the library attempted to set the IRQ target state of the P0 peripheral.
- FOTA download:
  - Added an optional progress event (`FOTA_DOWNLOAD_EVT_PROGRESS`) that informs the user of the library how many bytes have been downloaded.
  - Fixed a bug where the library continued downloading even if writing to the DFU target failed.
  - Implemented a mechanism to retry downloads if a socket error occurs.
- AWS FOTA:
  - Added functionality to resume jobs that are marked as being in progress, which ensures a more robust FOTA operation through AWS IoT jobs.
  - Added offset reporting through the `statusDetails` field in an AWS IoT job, which makes it possible to track the progress of a FOTA operation more precisely.
  - Removed the unused `app_version` parameter from the `aws_fota_init()` function.
  - Inversed the interpretation of the return value of `aws_fota_mqtt_evt_handler()` . 0 now indicates success, and no further handling is required. 1 indicates that further processing is required by the `mqtt_evt_handler()` that called `aws_fota_mqtt_evt_handler()` .
- nRF Cloud library:
  - Removed the button/switch pairing method.
  - Added functionality to handle the device configuration in the device shadow.
- LwM2M carrier:
  - Updated to version 0.8.1.
- at_host (`lib/at_host`):
  - Changed the default line ending from `CR` to `LF` in Kconfig to support sending SMS.
- Moved the following libraries from `drivers/` to `lib/` :
  - AT command interface
  - `lte_link_control`

Updated drivers

- Moved the following drivers from `drivers/` to `drivers/gps/` :
  - `gps_sim`
  - `nrf9160_gps`

BSD library

- Updated the BSD library to version 0.6.1.

nRF5340

This release demonstrates a dual-core solution with the Bluetooth LE Controller running on the network core and the Bluetooth LE Host and application running on the application core of the nRF5340.

Both Nordic Semiconductor's nRF Bluetooth LE Controller and Zephyr's Bluetooth LE Controller have been ported to run on the network core (nrf5340_dk_nrf5340_cpunet). The application core (nrf5340_dk_nrf5340_cpuapp) can run Bluetooth LE samples from both the nRF Connect SDK and Zephyr.

- Added the following sample:
  - Radio Test - runs on the network core and demonstrates how to configure the radio in a specific mode and then test its performance. This sample was ported from the nRF5 SDK.
- Added support for the nRF5340 PDK board (PCA10095) with board targets nrf5340_dk_nrf5340_cpunet and nrf5340_dk_nrf5340_cpuapp.
- Updated nrfx to support nRF5340.
- Added NFC support.

Common libraries, drivers, and samples

- Added the following libraries:
  - Hardware flash write protection - can be used to protect flash areas from writing. This library was extracted from the Immutable bootloader sample.
  - `lib\fatal_error` - overrides the default fatal error handling in Zephyr. By default, all samples perform a system reset if a fatal error occurs.

Updated samples and applications

- Immutable bootloader:
  - Moved the provisioning data (slot sizes/addresses and public keys) to one-time programmable (OTP) memory for nRF9160 devices.
  - Implemented invalidation of public keys.

Updated libraries

- Firmware information:
  - Renamed ABIs to EXT_APIs.
  - Restructured the `fw_info` structure:
    - Renamed the fields `firmware_size`, `firmware_address`, and `firmware_version` to `size`, `address`, and `version`.
    - Added a field to invalidate the structure.
    - Added reserved fields for future use.

        - EXT_APIs are now in a list at the end of the structure, instead of being available behind a function call.
        - EXT_APIs can now be requested by adding a request structure to a list after the EXT_API list itself.
    - Updated how EXT_API requests are checked. Requests are now checked against EXT_APIs by the bootloader before booting the image.
    - Added two new allowed offsets for the struct: 0x0 and 0x1000 bytes.
    - Removed `memeq()` in favor of regular `memcmp()`.
    - Renamed `__ext_api()` to `EXT_API()`, because names starting with `__` are reserved for the compiler.
    - Added new configuration options `CONFIG_*_EXT_API_REQUIRED` and `CONFIG_*_EXT_API_ENABLED` for, respectively, users and providers of EXT_APIs.
- DFU target:
    - Added the configuration option **`CONFIG_DFU_TARGET_MCUBOOT_SAVE_PROGRESS`**, which uses Zephyr's Settings subsystem. When this option is enabled, the write progress of an MCUboot style upgrade is stored, so that the progress is retained when the device reboots.
    - Fixed a bug where **`dfu_target_done()`** logged the error message `unable to deinitialize dfu resource` when no target was initialized.
- Moved the following libraries from `drivers/` to `lib/`:
    - Hardware flash write protection
    - NFC Reader ST25R3911B
    - `adp536x`
    - `flash_patch`

Crypto

- Added low-level cryptographic test suite using NIST, RFCs, and custom test vectors.
- nrf_cc310_mbedcrypto library/nrf_cc310_platform library v0.9.2:
    - Fixed power-efficiency issues.
    - Added experimental use of CryptoCell interrupt instead of busy-waits.
- CC310 hardware driver:
    - Added support for CryptoCell interrupt.

nRF Bluetooth LE Controller

- Updated the nRF BLE Controller libraries:
    - Removed version numbers for the libraries.
    - Added preliminary support for the S140 variant with the nRF5340 device. The Bluetooth LE Controller for nRF5340 supports the same feature set as its nRF52 Series counterpart.
    - Moved some APIs to MPSL. The library must now be linked together with MPSL.
    - Made Data Length Extensions a configurable feature.

  - Fixed an issue where an assert could occur when receiving a packet with a CRC error after performing a data length procedure on Coded PHY.

For details, see the Changelog.

Multi-Protocol Service Layer (MPSL)

- Updated the MPSL libraries:
  - Removed version numbers for the libraries.
  - Added a library version with preliminary support for the nRF5340 device. The feature set is the same as in the MPSL library for nRF52.

For details, see the Changelog.

Subsystems

Bluetooth Low Energy

- Added the following samples:
  - Bluetooth: Peripheral GATT Discovery Manager - demonstrates how to use the GATT Discovery Manager.
  - Bluetooth: LLPM - showcases the proprietary Low Latency Packet Mode (LLPM) extension.
- Updated the Bluetooth LE samples:
  - Enabled stack protection, assertions, and logging by default.
  - Modified the samples to use the synchronous `bt_enable()` function.
- Nordic UART Service (NUS) Client, GATT Battery Service (BAS) Client, and GATT DFU SMP Service Client:
  - Fixed an issue where it was not possible to subscribe to the service notifications more than once.
- Updated the Bluetooth: Central UART sample to handle data packets that are longer than 212 bytes. Enabled UART flow control to avoid data loss.
- Enabled UART flow control in the Bluetooth: Peripheral UART sample to avoid data loss.
- Changed the Bluetooth: Throughput sample to prevent it from running Bluetooth LE scanning and advertising in parallel. The feature to establish a connection in both master and slave role at the same time is not supported by the Zephyr Bluetooth LE Host.
- Scanning module:
  - Added an option to update the initial connection parameters.
- GATT Discovery Manager:
  - Fixed an issue where service or characteristic allocation failed, but the returned pointer was not checked before accessing the data it pointed to.

NFC

- Added the following samples:
  - NFC: TNEP Tag and NFC: TNEP Poller - demonstrate how to use the Tag NDEF Exchange Protocol (TNEP).
  - NFC: System OFF - demonstrates how to wake the device from System OFF mode using NFC. This sample was ported from the nRF5 SDK.
- Added the following libraries:
  - Capability Containers file parser - reads and parses the Capability Container file that can be found in the Type 4 Tag.
  - NFC T4T High Level Procedure - performs the NDEF detection procedure for the Type 4 Tag.
  - Tag NDEF Exchange Protocol for NFC Tag Device - implements the Tag NDEF Exchange Protocol (TNEP) for a Tag device.
  - Tag NDEF Exchange Protocol for NFC Poller Device - implements the Tag NDEF Exchange Protocol (TNEP) for a Poller device.
- Updated the NFC samples to enable stack protection, assertions, and logging by default.
- Extended the NFC: Tag Reader sample with parsing and printing of the Type 4 Tag content.
- Moved the NFC Platform implementation to the fw-nrfconnect-nrf repository. See Integration notes.

Multi-Protocol Service Layer (MPSL)

- Added MPSL as a new subsystem. It integrates MPSL into the nRF Connect SDK environment.
- Added the following sample:
  - MPSL timeslot - demonstrates how to use MPSL and basic MPSL Timeslot functionality.

Setting storage

- Reduced the default partition size for the settings storage from 24 kB (0x6000) to 8 kB (0x2000). This leaves more flash space to the application.

nRF Desktop

- Added a `ble_qos` module to maintain channel maps.

Build system

- Fixed a bug where a user-defined HEX file that was provided in the static configuration of the Partition Manager was not included in the merge operation.

nrfx

- Updated to v2.1.0. For details, see the changelog.

MCUboot

- Updated to include new features from upstream:
  - New downgrade prevention feature (available when the overwrite-based image update strategy is used)
  - New swap method that removes the need for a scratch partition
  - Bug fixes

  See the MCUboot release notes for more information. Note that not all features from v1.5.0 are included.

Documentation

- Added or updated documentation for the following samples:
  - nRF9160:
    - nRF9160: Cloud Client - added
    - nRF9160: GPS sockets and SUPL client library - added
    - nRF9160: HTTPS Client - added
    - nRF9160: Serial LTE Modem - added
    - nRF9160: nRF CoAP Client - updated
  - Bluetooth Low Energy:
    - Bluetooth: LLPM - added
    - Bluetooth: Peripheral GATT Discovery Manager - added
    - Bluetooth: Throughput - updated
  - NFC:
    - NFC: System OFF - added
    - NFC: TNEP Poller - added
    - NFC: TNEP Tag - added
  - Other:
    - Radio Test - added
    - MPSL timeslot - added
    - Immutable bootloader - updated
    - nRF Desktop - updated
- Added or updated documentation for the following libraries:
  - nRF9160:
    - AWS IoT - added
    - Cloud API - added
    - Modem key management - added
    - SMS subscriber - added
    - SUPL client - added

        - AT command interface - updated
        - AT command parser - updated
        - Download client - updated
    - Bluetooth Low Energy:
        - Bluetooth Mesh provisioning handler for Nordic DKs - added
        - GATT Latency Client - added
        - GATT Latency Service - added
        - Nordic UART Service (NUS) shell transport - updated
    - NFC:
        - Capability Containers file parser - added
        - NFC T4T High Level Procedure - added
        - Tag NDEF Exchange Protocol for NFC Poller Device - added
        - Tag NDEF Exchange Protocol for NFC Tag Device - added
    - Other:
        - Bootloader crypto - added
        - Bootloader firmware validation - added
        - Hardware flash write protection - added
        - DFU target - updated
        - Firmware information - updated
        - Partition Manager - updated
- Added or updated the following documentation:
    - Getting started:
        - Installing the nRF Connect SDK manually - restructured parts of the content
        - Building and programming a sample application - restructured the content and added information about building on the command line
        - Modifying a sample application - updated the content and added information about configuring an application
    - User guides:
        - Working with nRF5340 - added
        - Working with Thingy:91 - added
        - Bluetooth LE Controller - added
        - Multi-image builds - updated with content that was removed from the Zephyr fork
    - nrfxlib:
        - BSD library - extended and restructured the content
        - MPSL - added
        - Integration with applications - updated to match current version of the nRF Bluetooth LE Controller

Known issues

nRF9160

- The `nrf_send()` function in the BSD library might be blocking for several minutes, even if the socket is configured for non-blocking operation. The behavior depends on the cellular network connection.
- The nRF9160: GPS sockets and SUPL client library sample stops working if SUPL client support is enabled, but the SUPL host name cannot be resolved. As a workaround, insert a delay (`k_sleep()`) of a few seconds after the `printf` on line 294 in `main.c`.
- The nRF9160: Asset Tracker sample might show up to 2.5 mA current consumption in idle mode with `CONFIG_POWER_OPTIMIZATION_ENABLE=y`.
- The SEGGER Control Block cannot be found by automatic search by the RTT Viewer/Logger. As a workaround, set the RTT Control Block address to 0 and it will try to search from address 0 and upwards. If this does not work, look in the `builddir/zephyr/zephyr.map` file to find the address of the `_SEGGER_RTT` symbol in the map file and use that as input to the viewer/logger.
- nRF91 fails to receive large packets (over 4000 bytes).
- nrf_connect fails if called immediately after initialization of the device. A delay of 1000 ms is required for this to work as intended.

Subsystems

Bluetooth Low Energy

- Bluetooth LE cannot be used in a non-secure application, for example, an application built for the nrf5340_dk_nrf5340_cpuappns board. Use the nrf5340_dk_nrf5340_cpuapp board instead.
- The Bluetooth: Peripheral HIDS keyboard sample cannot be used with the nRF BLE Controller because the NFC subsystem does not work with the controller library. The library uses the MPSL Clock driver, which does not provide an API for asynchronous clock operation. NFC requires this API to work correctly.
- When the Bluetooth: Peripheral HIDS mouse sample is used with the Zephyr Bluetooth LE Controller, directed advertising does not time out and the regular advertising cannot be started.
- The Bluetooth: Central HIDS sample cannot connect to a peripheral that uses directed advertising.
- Bluetooth Low Energy peripheral samples are unstable in some conditions (when pairing and bonding are performed and then disconnections/re-connections happen).
- When running the Bluetooth: Central DFU SMP sample, the `CONFIG_BT_SMP` configuration must be aligned between this sample and the Zephyr counterpart (SMP Server Sample). However, security is not enabled by default in the Zephyr sample.
- The central samples (Bluetooth: Central UART, Bluetooth: Central HIDS) do not support any pairing methods with MITM protection.
- On some operating systems, the nrf_desktop application is unable to reconnect to a host.
- central_uart: A too long 212-byte string cannot be handled when entered to the console to send to peripheral_uart.

Bootloader

- Building and programming the immutable bootloader (see Secure bootloader chain) is not supported in SEGGER Embedded Studio.
- The immutable bootloader can only be used with the following boards:
  - nrf52840_pca10056
  - nrf9160_pca10090

  It does not work properly on nRF51 and nRF53.

NFC

- The NFC: TNEP Poller and NFC: Tag Reader samples cannot be run on the nRF5340 PDK. There is an incorrect number of pins defined in the MDK files, and the pins required for NFC Reader ST25R3911B cannot be configured properly.
- NFC tag samples are unstable when exhaustively tested (performing many repeated read and/or write operations). NFC tag data might be corrupted.

Build system

- It is not possible to build and program nRF9160: Secure Partition Manager and the application individually.

nrfxlib

- In the BSD library, the GNSS sockets implementation is experimental.

MDK (part of nrfx)

- For nRF5340, the pins **P1.12** to **P1.15** are unavailable due to an incorrect pin number definition in the MDK.

MCUboot

- The MCUboot recovery feature using the USB interface does not work.

In addition to the known issues above, check the current issues in the official Zephyr repository, since these might apply to the nRF Connect SDK fork of the Zephyr repository as well. To get help and report issues that are not related to Zephyr but to the nRF Connect SDK, go to Nordic's DevZone.