s122_nrf52 migration document

Introduction to the s122_nrf52 migration document

About the document

This document describes how to migrate to new versions of SoftDevice S122. The release notes for this SoftDevice should be read in conjunction with this document.

For each version, we have the following sections:

- "Required changes" describes the changes that need to be done in the application when migrating from an older version of the SoftDevice.
- "New functionality" describes how to use new features and functionality offered by this version of the SoftDevice. **Note:** Not all new functionality may be covered; the release notes will contain a full list of new features and functionality.

Each section describes how to migrate to a given version from the previous version. If you are migrating to the current version from the previous version, follow the instructions in that section. To migrate between versions that are more than one version apart, follow the migration steps for all intermediate versions in order.

Example: To migrate from version 5.0.0 to version 5.2.0, first follow the instructions to migrate to version 5.1.0 from version 5.0.0, then follow the instructions to migrate to version 5.2.0 from version 5.1.0.

s122_nrf52_8.0.0

This section compares the major changes from SoftDevice S132 v7.0.1 to the new variant SoftDevice S122 v8.0.0. Due to API changes between these versions, applications have to be to be recompiled.

Note on different device compatibility

SoftDevice S132 v7.0.1 is compatible with nRF52810 and nRF52832.

SoftDevice S122 v8.0.0 is compatible with nRF52833 and nRF52820.

Required changes

Removal of peripheral features

The major differentiating feature of the new S122 variant is the lack of a peripheral role and Advertising Extension support. This variant is optimized for applications utilizing only the central role. The changes in this section are removals of advertising, peripheral, and related features.

Peripheral and advertising roles

The following types have been removed. The * denotes a wildcard.

- `ble_gap_adv_*_t`
- `ble_gap_evt_adv_set_terminated_t`
- `ble_gap_evt_scan_req_report_t`
- `ble_gap_evt_sec_info_request_t`
- `ble_gap_opt_local_conn_latency_t`
- `ble_gap_opt_slave_latency_disable_t`

The following functions have been removed. The * denotes a wildcard.

- `sd_ble_gap_adv_*`
- `sd_ble_gap_ppcp_*`
- `sd_ble_gap_sec_info_reply`

The following constants have been removed. The * denotes a wildcard.

- `BLE_ERROR_GAP_DISCOVERABLE_WITH_WHITELIST`
- `BLE_GAP_ADDR_TYPE_ANONYMOUS`
- `BLE_GAP_ADV_*`
- `BLE_GAP_EVT_SCAN_REQ_REPORT`
- `BLE_GAP_EVT_SEC_INFO_REQUEST`
- `BLE_GAP_OPT_LOCAL_CONN_LATENCY`
- `BLE_GAP_OPT_SLAVE_LATENCY_DISABLE`
- `BLE_GAP_ROLE_COUNT_PERIPH_DEFAULT`
- `BLE_GAP_ROLE_PERIPH`
- `BLE_GAP_SCAN_BUFFER_EXTENDED_MAX`
- `BLE_GAP_SCAN_BUFFER_EXTENDED_MAX_SUPPORTED`
- `BLE_GAP_SCAN_BUFFER_EXTENDED_MIN`
- `BLE_GAP_TX_POWER_ROLE_ADV`
- `SD_BLE_GAP_PPCP_*`
- `SD_BLE_GAP_SEC_INFO_REPLY`

The following fields have been removed:

- `ble_gap_cfg_role_count_t::adv_set_count`
- `ble_gap_cfg_role_count_t::periph_role_count`
- `ble_gap_evt_connected_t::adv_data`
- `ble_gap_evt_connected_t::adv_handle`

The following fields are not used, but have not been removed:

- `ble_gap_scan_params_t::extended`
- `ble_gap_scan_params_t::report_incomplete_evts`

Advertising extensions

The following symbols have been removed:

- `BLE_GAP_ADV_DATA_STATUS_INCOMPLETE_MISSED`
- `BLE_GAP_ADV_DATA_STATUS_INCOMPLETE_MORE_DATA`
- `BLE_GAP_ADV_DATA_STATUS_INCOMPLETE_TRUNCATED`

The following fields are no longer used, but have not been removed:

- `ble_gap_evt_adv_report_t::aux_pointer`

Extended RC calibration

The following symbols have been removed:

- BLE_COMMON_OPT_EXTENDED_RC_CAL
- ble_common_opt_extended_rc_cal_t
- ble_common_opt_t::extended_rc_cal

Removal of Connection-Oriented Channels in LE Credit Based Flow Control Mode

This SoftDevice variant does not support Connection-Oriented Channels in LE Credit Based Flow Control Mode. The following has been removed to reflect this:

- The header file l2cap.h
- The following symbols. The * denotes a wildcard.
    - BLE_L2CAP_*
    - ble_l2cap_*
    - SD_BLE_L2CAP_*
    - sd_ble_l2cap_*
- The configuration field ble_conn_cfg_t::params.l2cap_conn_cfg

Removal of LE Data Packet Length Extension (DLE)

This SoftDevice variant does not support DLE. The following have been removed to reflect this:

- SD_BLE_GAP_DATA_LENGTH_UPDATE
- BLE_GAP_EVT_DATA_LENGTH_UPDATE_REQUEST
- BLE_GAP_EVT_DATA_LENGTH_UPDATE
- ble_gap_data_length_params_t
- ble_gap_data_length_limitation_t
- ble_gap_evt_data_length_update_request_t
- ble_gap_evt_data_length_update_t
- sd_ble_gap_data_length_update

RSSI

7.0.1	8.0.0
sd_ble_gap_rssi_start(x, y, z)	sd_ble_gap_qos_start(BLE_GAP_RSSI, &(ble_gap_qos_params_t){.rssi={.conn_handle=x, .threshold_dbm=y, .skip_count=z}})
sd_ble_gap_rssi_stop(x)	sd_ble_gap_qos_stop(BLE_GAP_RSSI, &(ble_gap_qos_params_t){.rssi={.conn_handle=x}})

QoS channel survey

7.0.1	8.0.0
`sd_ble_gap_qos_channel_survey_start(x)`	`sd_ble_gap_qos_start(BLE_GAP_QOS_CHANNEL_SURVEY, &(ble_gap_qos_params_t){.channel_survey={.interval_us=x}})`
`sd_ble_gap_qos_channel_survey_stop()`	`sd_ble_gap_qos_stop(BLE_GAP_QOS_CHANNEL_SURVEY, NULL)`

Connection event trigger

7.0.1	8.0.0
`sd_ble_gap_conn_evt_trigger_start(conn_handle, p_params)`	`sd_ble_gap_evt_trigger_start(BLE_GAP_LL_ROLE_CONN, conn_handle, p_params)`
`sd_ble_gap_conn_evt_trigger_stop(conn_handle)`	`sd_ble_gap_evt_trigger_stop(BLE_GAP_LL_ROLE_CONN, conn_handle)`
`ble_gap_conn_event_trigger_t`	`ble_gap_event_trigger_t`

Changes to scanner timing fields

Scan windows and intervals are now given in microseconds and include the post-processing overhead. Previously, scan windows and intervals were defined in terms of units of 625 s and did not include the post-processing.

7.0.1	8.0.0
`ble_gap_scan_params_t::interval`	`ble_gap_scan_params_t::interval_us`
`ble_gap_scan_params_t::window`	`ble_gap_scan_params_t::window_us`
`BLE_GAP_SCAN_INTERVAL_MIN`	`BLE_GAP_SCAN_INTERVAL_US_MIN`
`BLE_GAP_SCAN_INTERVAL_MAX`	`BLE_GAP_SCAN_INTERVAL_US_MAX`
`BLE_GAP_SCAN_WINDOW_MIN`	`BLE_GAP_SCAN_WINDOW_US_MIN`
`BLE_GAP_SCAN_WINDOW_MAX`	`BLE_GAP_SCAN_WINDOW_US_MAX`

Removal of unused defines

7.0.1	8.0.0

BLE_GAP_DISC_MODE_NOT_DISCOVERABLE	<removed>
BLE_GAP_DISC_MODE_LIMITED	<removed>
BLE_GAP_DISC_MODE_GENERAL	<removed>

New return code from APIs

The following APIs may now return with error BLE_ERROR_UNSUPPORTED_REMOTE_FEATURE when the peer is known to not support the corresponding feature.

- sd_ble_gap_encrypt()
- sd_ble_gap_phy_update()

Removed old flash protection API

sd_flash_protect(), has been removed in this SoftDevice. This API is superseded by the sd_protected_register_write() API in conjunction with the ACL peripheral.

New functionality

Event length check

The SoftDevice by default prevents the application from changing the connection state to utilize more than the configured ble_gap_conn_cfg_t::event_length. This helps set up a multi-link scenario without scheduling conflicts. The feature can be disabled using the following option:

```
sd_ble_opt_set(BLE_GAP_OPT_ENABLE_EVT_LEN_CHECK, &(ble_opt_t){.gap_opt.enable_evt_len_check.enable = 0});
```

This option can be used to set up multiple connections with ble_gap_conn_cfg_t::event_length set to BLE_GAP_EVENT_LENGTH_2MBPS_PHY_MIN. If the event length check is not disabled, the SoftDevice prevents the application from establishing a connection when the connection is configured to use an event length of BLE_GAP_EVENT_LENGTH_2MBPS_PHY_MIN. Before the connection is switched to 2 Mbps PHY, scheduling conflicts between links can occur. For more information on scheduling, see the SoftDevice Specification for this variant.

Connection event QoS report

The connection event QoS report gives the application the number of CRC failures for each BLE connection event. This enables the application to make an informed decision when selecting a channel map.

Usage

```
sd_ble_gap_qos_start(BLE_GAP_QOS_CONN_EVENT, NULL);

[...]

/* In event handling code, with evt from sd_ble_evt_get: */
switch (evt.header.evt_id)
{
  case BLE_GAP_EVT_QOS_CONN_EVENT_REPORT:
    /* Application specific handling of the event parameters: */
    evt.evt.conn_handle;
    evt.evt.params.qos_conn_event_report.event_counter;
    evt.evt.params.qos_conn_event_report.ch_index;
    evt.evt.params.qos_conn_event_report.crc_ok_count;
    evt.evt.params.qos_conn_event_report.crc_error_count;
    evt.evt.params.qos_conn_event_report.rx_timeout;
    break;
}
```

Radio notifications

Two new shorter distances for radio notifications are available.

- `NRF_RADIO_NOTIFICATION_DISTANCE_200US`
- `NRF_RADIO_NOTIFICATION_DISTANCE_420US`

Access to USB power handling registers

The SoftDevice provides new APIs allowing the application to enable or disable USB power interrupts. It is also now possible to read the value of the USB supply status register.

API Updates

Four new APIs have been added:

- `sd_power_usbpwrrdy_enable()` - Enable or disable the USB power ready event.

  When enabled, the `NRF_EVT_POWER_USB_POWER_READY` event will be raised when USB 3.3 V supply is ready.

- `sd_power_usbdetected_enable()` - Enable or disable the USB power detected event.

  When enabled, the `NRF_EVT_POWER_USB_DETECTED` event will be raised when voltage supply is detected on the VBUS pin.

- `sd_power_usbremoved_enable()` - Enable or disable the USB power removed event.

  When enabled, the `NRF_EVT_POWER_USB_REMOVED` event will be raised when voltage supply removed from the VBUS pin.

- `sd_power_usbregstatus_get()` - Get the USB supply status register content.

Scanner and initiator event trigger

The SoftDevice can now be configured to trigger a PPI just before a scan or connection initiation. These are configured independently and in addition to the connection interval event covered in the “Required changes” section.

Event triggering on scan start:

```
sd_ble_gap_evt_trigger_start(BLE_GAP_LL_ROLE_SCAN, NULL, ...);
```

Event triggering on connection initiation:

```
sd_ble_gap_evt_trigger_start(BLE_GAP_LL_ROLE_INIT, NULL, ...);
```

Other

The Memory Watch Unit is not used by this SoftDevice variant. The SoftDevice will not prevent the application from accessing SoftDevice memory improperly. Improper access can result in SoftDevice malfunction.