S110 nRF51822

Bluetooth low energy

SoftDevice Specification v0.6

Key Features

- *Bluetooth* 4.0 compliant low energy single-mode protocol stack
    - Link layer
    - L2CAP, ATT, and SM protocols
    - GATT, GAP, and L2CAP
    - Peripheral and broadcaster roles
    - GATT Client and Server
    - Full SMP support including MITM and OOB pairing
- Complementary nRF518 SDK including *Bluetooth*® profiles and example applications
- Memory isolation from Application for protocol implementation robustness and security
- Thread-safe supervisor-call based API
- Asynchronous, event-driven behavior
- No RTOS dependency
    - A RTOS of your choice can be used
- Low link-time dependencies
    - Standard ARM Cortex™-M0 project configuration for application development
- Support for non-concurrent multi-protocol operation
    - Alternate protocol stack running in application space

Applications

- Computer peripherals and I/O devices
    - Mouse
    - Keyboard
    - Multi-touch trackpad
- Interactive entertainment devices
    - Remote control
    - 3D glasses
    - Gaming controller
- Personal Area Networks
    - Health and fitness sensor and monitor devices
    - Medical devices
    - Key-fobs and wrist watch
- Remote control toys

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.

Life support applications

Nordic Semiconductor's 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.

Contact details

For your nearest dealer, please see http://www.nordicsemi.com.
Information regarding product updates, downloads, and technical support can be accessed through your My Page account on our homepage.

Main office: Otto Nielsens veg 12
7052 Trondheim
Norway

Phone: +47 72 89 89 00
Fax: +47 72 89 89 89

Mailing address: Nordic Semiconductor
P.O. Box 2336
7004 Trondheim
Norway

RoHS and REACH statement

Nordic Semiconductor's products meet the requirements of Directive 2002/95/EC of the European Parliament and of the Council on the Restriction of Hazardous Substances (RoHS) and the requirements of the REACH regulation (EC 1907/2006) on Registration, Evaluation, Authorization and Restriction of Chemicals. The SVHC (Substances of Very High Concern) candidate list is continually being updated. Complete hazardous substance reports, material composition reports and latest version of Nordic's REACH statement can be found on our website www.nordicsemi.com.

Revision History

Date	Version	Description
September 2012	0.6	

Contents

1 Introduction

S110 is a *Bluetooth*® low energy peripheral protocol stack solution. It integrates LE controller and host, and provides a full and flexible API for building *Bluetooth* low energy System on Chip (SoC) solutions with Nordic Semiconductor nRF51 ICs.

This document contains information about the SoftDevice features and performance.

Note: The SoftDevice features and performance are subject to change between revisions of this document. See ***chapter 9 on page 21*** for more information.

1.1 Required reading

The nRF51822 Product Specification and the nRF51 Series Reference Manual are essential developer resources for *Bluetooth*® low energy solutions from Nordic Semiconductor. The software architecture chapter in the nRF51 Series Reference Manual is essential reading for understanding the resource usage and performance related chapters of this document.

Knowledge of *Bluetooth* specification, Ver. 4.0, Volumes 1, 3, 4 and 6 is required to use the S110 correctly and for understanding the terminology used within this document.

1.2 Writing conventions

This SoftDevice Specification follows a set of typographic rules to ensure that the document is consistent and easy to read. The following writing conventions are used:

- Command, event names, and bit state conditions are written in `Lucida Console`.
- Pin names and pin signal conditions are written in **`Consolas.`**
- File names and User Interface components are written in **bold**.
- Internal cross references are italicized and written in ***semi-bold.***
- Placeholders for parameters are written in *italic regular font*. For example, a syntax description of SetChannelPeriod will be written as: SetChannelPeriod(*ChannelNumber, MessagingPeriod*).
- Fixed parameters are written in regular text font. For example, a syntax description of SetChannelPeriod will be written as: SetChannelPeriod (0, Period).

2 Product overview

The S110 SoftDevice is a precompiled and linked binary software implementing a *Bluetooth* 4.0 low energy (BLE) protocol stack. The S110 is compatible with selected nRF518xx System on Chip (SoC) devices. The Application Programming Interface (API) is offered as a standard C language interface giving the application complete compiler and linker independence from the SoftDevice implementation. From an application development point of view, a SoftDevice enables the application programmer to develop their code as a standard ARM® Cortex™-M0 project without the need to integrate with proprietary chip-vendor software frameworks. This means that any ARM® Cortex™-M0 compatible toolchain can be used to develop *Bluetooth* low energy applications with the S110 SoftDevice.

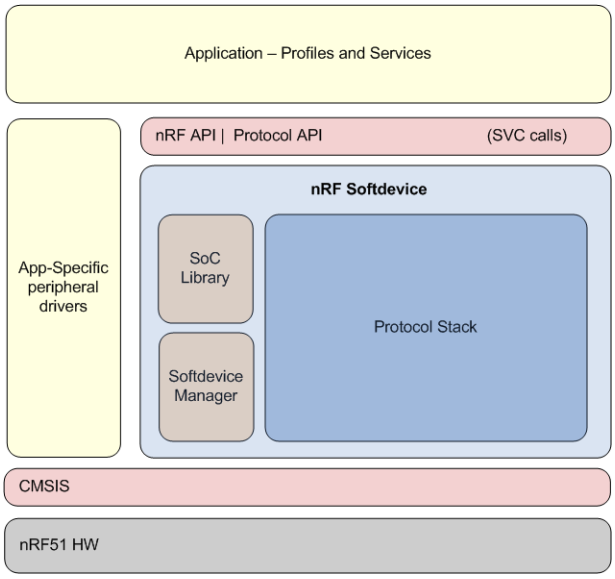


Figure 1 *System on Chip application with the SoftDevice*

The S110 SoftDevice can be programmed onto compatible nRF518xx devices in development and in production. The S110 SoftDevice will be updated with additional features and/or fixed issues if needed. Previous supported versions of the S110 SoftDevice will remain available after updates so products do not need to be re-qualified on release of updates if the previous version is sufficiently feature-complete for your product.

This specification outlines features and support for the complete S110 SoftDevice. Alpha and Beta versions may not have support for all features. To find information on any limitations or omissions, the S110 SoftDevice release notes will contain a detailed summary of the release status.

2.1 Multi-protocol support

The S110 SoftDevice supports non-concurrent multi-protocol implementations. This means a proprietary 2.4 GHz protocol can be implemented in the application program area. This protocol can access all hardware resources when the S110 SoftDevice is in a disabled state. When the S110 SoftDevice is disabled, the proprietary 2.4 GHz protocol can run in application program space.

3 *Bluetooth* low energy Protocol Stack

The *Bluetooth* 4.0 compliant low energy (BLE) Host and Controller embedded in the S110 SoftDevice are fully qualified with multi-role support (peripheral and broadcaster). The Application Program Interface (API) is defined above the Generic Attribute Protocol (GATT), Generic Access Profile (GAP), and Logical Link Control and Adaptation Protocol (L2CAP). The S110 SoftDevice supports approved *Bluetooth* low energy and proprietary GATT profile implementations.

The nRF518 Software Development Kit (SDK) completes the BLE protocol stack with Service and Profile implementations. Single-mode System on Chip (SoC) applications are enabled by the full BLE protocol stack and nRF518xx integrated circuit (IC).

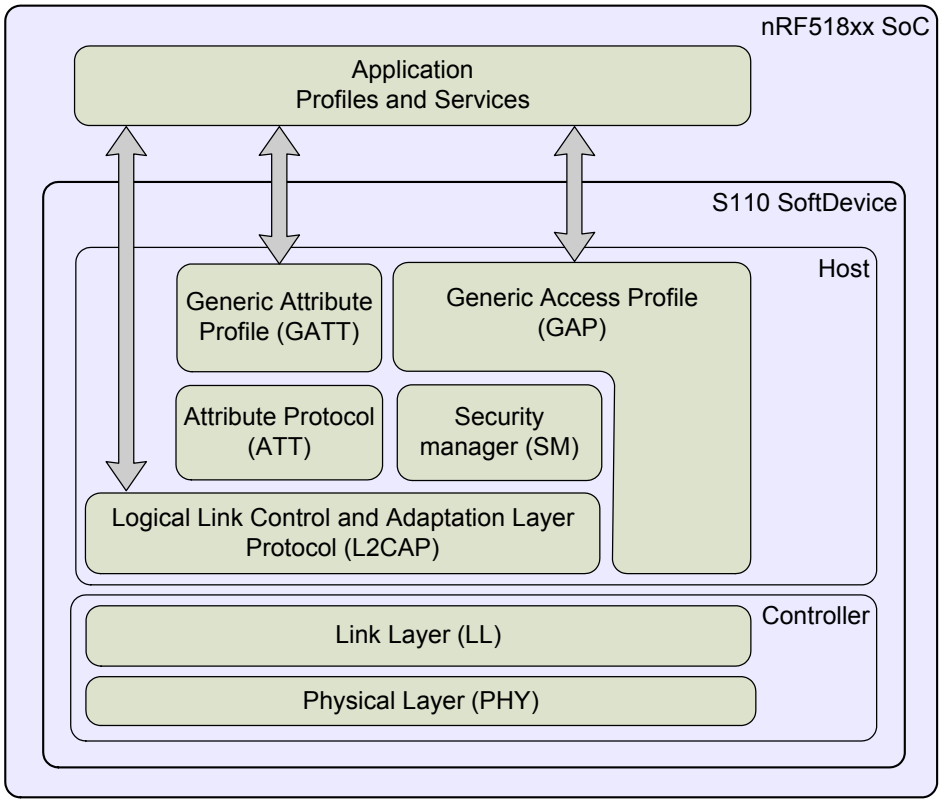


Figure 2 *LE stack architecture*

3.1 Profile and service support

The peripheral behavior for all adopted profiles is supported by the S110 SoftDevice at the time of publishing.

The following table shows the **Profiles and corresponding Services** supported by the S110 SoftDevice.

Adopted Profile	Adopted Services	Supported
HID over GATT	HID	YES
	Battery	
	Device Information	
Heart Rate Monitor	Heart Rate	YES
	Device Information	
Proximity	Link Loss	YES
	Immediate Alert	
	TX Power	
Blood Pressure	Blood pressure	YES
Health Thermometer	Health Thermometer	YES
Glucose	Glucose	YES
Phone Alert Status	Phone Alert Status	YES
Alert Notification	Alert Notification	YES
Time	Current Time	YES
	Next DST Change	
	Reference Time Update	
Find Me	Immediate Alert	YES
Cycling speed & cadence	Cycling speed & cadence Device information	YES
Running speed & cadence	Running speed & cadence Device information	YES

***Table 1** Adopted Profile and Service support*

3.2 *Bluetooth* low energy features

The BLE protocol stack in the S110 SoftDevice has been designed to provide an abstract, but flexible, interface for application development for *Bluetooth* SMART devices. GAP, GATT, SM, and L2CAP are implemented in the SoftDevice and managed through the API. Procedures and modes used by profiles and defined in the *Bluetooth* 4.0 specification are implemented in the stack for the developer; for example, discoverable and connection modes, and pairing and bonding procedures.

The BLE API is consistent across *Bluetooth* role implementations; features in common have the same interface.

In the following sections, the features of the BLE stack in the S110 SoftDevice are identified.

3.3 API

Feature	Notes
Interface to: GATT/GAP/L2CAP	Consistency between APIs including shared data formats
GATT DB population & access	Full flexibility to populate, but no dynamic (i.e. attribute removal) support yet
Thread-safe, asynchronous, and event driven	Minimizes exposure to concurrency issues
Vendor-specific (128 bit) UUIDs for proprietary profiles	Compact, fast, and memory efficient management of 128 bit UUIDs
Non-concurrent multi-protocol	
Packet flow control	Zero-copy buffer management

3.4 Generic Access Profile (GAP)

Feature	Notes
Multi-role: Peripheral & Broadcaster	
Limited and general discoverable modes	
Multiple bond support	Keys and peer information stored in application space No limitations in stack implementation
User-defined Advertising data	Full control over advertising and scan response data for the application
Security mode 1: Levels 1, 2, and 3	
Resolvable address whitelisting based on IRK	Synchronous and low power solution for BLE enhanced privacy

3.5 Generic Attribute Profile (GATT)

Features	Notes
Full GATT Server	
Server deferred operations: R/W characteristic value R/W descriptors	Enables control points Enables freshest data Enables GAP authorization
Full GATT Client	Flexible data management options for packet transmission with either fine control or abstract management
Implemented GATT Sub-procedures	Discover all Primary Services Discover Primary Service by Service UUID Find included Services Discover All Characteristics of a Service Discover Characteristics by UUID Discover All Characteristic Descriptors Read Characteristic Value Read using Characteristic UUID Read Long Characteristic Values Write Without Response Write Characteristic Value Notifications Indications Read Characteristic Descriptors Read Long Characteristic Descriptors Write Characteristic Descriptors

3.6 Security Manager (SM)

Feature	Notes
Lightweight key storage for reduced NV memory requirements	
Authenticated MITM (Man in the middle) protection	
Pairing methods: Just works, Passkey Entry, and Out of Band	

3.7 Attribute Protocol (ATT)

Feature	Notes
Server protocol	
Client protocol	

3.8 Logical Link Control and Adaptation Layer Protocol (L2CAP)

Feature	Notes
27 byte MTU size	
Dynamically allocated channels	

3.9 Controller, Link Layer (LL)

Feature	Notes
Slave role	
Slave connection update	
27 byte MTU	
Encryption	

3.10 Proprietary features

Feature	Notes
TX Power control through API	
Window Limiting	
Application Latency	
Resolvable address whitelisting based on IRK	Synchronous and low power solution for BLE enhanced privacy

4 SoC library

The following features are in place to ensure the Application and SoftDevice coexist with safe sharing of common SoC resources.

Feature	Notes
`nrf_mutex_` API	Atomic mutex API without the application needing to disable global interrupts.
`nrf_nvic_` API	Gives the application access to all NVIC features without corrupting SoftDevice configurations.
`nrf_rand_` API	Gives access to the true random number generator.
`nrf_power_` API	Safe power configuration interface.
`nrf_clock_` API	Safe clock configuration interface.
`nrf_wait_for_app_event`	Simple power management hook for the application.
`nrf_ppi_` API	Safe PPI access to dedicated Application PPI channels.

5 SoftDevice Manager

The following features enable the Application to manage the SoftDevice on a top level.

Feature	Notes
`nRF_SoftDevice_` API	Gives control of the SoftDevice state.

6 S110 Resource Requirements

The S110 SoftDevice uses on-chip resources including memory, system blocks, and peripheral blocks. These resources may be used while the SoftDevice is enabled or disabled. This chapter outlines how memory and hardware are used by the SoftDevice.

6.1 Memory resource map and usage

Figure 3 shows the memory map for program memory and RAM at run-time with the SoftDevice enabled. ***Table 2*** outlines the memory resource requirements, both when the SoftDevice is enabled, and disabled.

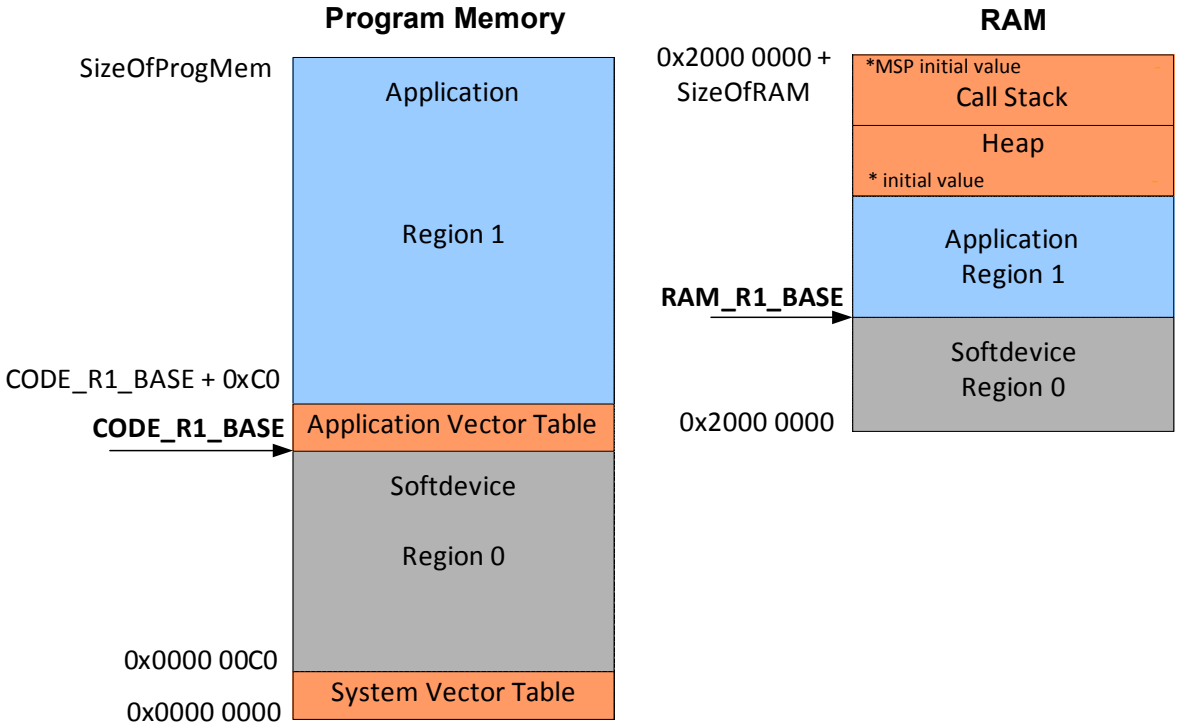


Figure 3 *Memory resource map*

Flash	S110 Enabled	S110 Disabled
Amount	128 kB	128 kB
CODE_R1_BASE	0x0002 0000	0x0002 0000

RAM	S110 Enabled	S110 Disabled
Amount	8 kB	0 kB
RAM_R1_BASE	0x2000 2000	0x2000 0000

Call stack	S110 Enabled	S110 Disabled
Maximum usage	1.5 kB (0x600)	0x00

Heap	S110 Enabled	S110 Disabled
Maximum allocated bytes	0 bytes (0x00)	0x00

***Table 2** S110 Memory resource requirements*

6.2 Hardware blocks and interrupt vectors

Table 3 defines access types used to indicate the availability of hardware blocks to the application. ***Table 4*** specifies access, per hardware block, both when the SoftDevice is enabled and disabled.

Access	Definition
Restricted	Used by the SoftDevice and outside the application sandbox. Application has restricted access via the SoftDevice API.
Blocked	Used by the SoftDevice and outside the application sandbox. Completely inaccessible to the application.
Open	Unused by the SoftDevice and in the application sandbox.

***Table 3** Hardware access definitions*

Peripheral protection and usage by SoftDevice				
ID	**Base address**	**Instance**	**Access (S110 enabled)**	**Access (S110 disabled)**
0	0x40000000	POWER	Restricted	Open
0	0x40000000	CLOCK	Restricted	Open
1	0x40001000	RADIO	Blocked	Open
2	0x40002000	UART0	Open	Open
3	0x40003000	SPIM0 / 2W0	Open	Open
4	0x40004000	SPIM1 / 2W1	Open	Open
		…		
6	0x40006000	Port 0 GPIOTE	Open	Open
7	0x40007000	ADC	Open	Open
8	0x40008000	TIMER0	Blocked	Open
9	0x40009000	TIMER1	Open	Open
10	0x4000A000	TIMER2	Open	Open
11	0x4000B000	RTC0	Blocked	Open
12	0x4000C000	TEMP	Open	Open
13	0x4000D000	RNG	Restricted	Open
14	0x4000E000	ECB	Restricted	Open
15	0x4000F000	CCM	Blocked	Open
15	0x4000F000	AAR	Blocked	Open
16	0x40010000	WDT	Open	Open
17	0x40011000	RTC1	Open	Open
18	0x40012000	QDEC	Open	Open
		…		
20	0x40014000	Software interrupt	Open	Open
21	0x40015000	Software interrupt	Open	Open
22	0x40016000	Software interrupt	Open	Open
23	0x40017000	Software interrupt	Blocked	Open
24	0x40018000	Software interrupt	Blocked	Open
25	0x40019000	Software interrupt	Blocked	Open
		…		
30	0x4001E000	NVMC	Open	Open
31	0x4001F000	PPI	Restricted	Open
NA	0x50000000	GPIO P0	Open	Open
NA	0xE000E100	NVIC	Restricted[1]	Open

1. Not sandboxed (protected). For robust system function, the application program must comply with the restriction and use the NVIC API for configuration when the SoftDevice is enabled.

Table 4 *Peripherals used by the SoftDevice*

6.3 PPI

When the SoftDevice is enabled, the PPI is restricted with only some PPI channels and groups available to the application. ***Table 5*** shows how channels and groups are assigned between the application and SoftDevice.

Note: All PPI channels are available to the application when the SoftDevice is disabled.

PPI channel allocation	S110 enabled	S110 disabled
Application	Channels 0 - 7	Channels 0 - 15
SoftDevice	Channels 8 - 15	-

PPI group allocation	S110 enabled	S110 disabled
Application	Channels 0 - 1	Groups 0 - 3
SoftDevice	Channels 2 - 3	-

Table 5 *PPI channel and group availability when PPI is restricted*

6.4 SVC number ranges

Table 6 shows which SVC numbers an application program can use, and which numbers are used by the SoftDevice.

Note: The SVC number allocation does not change with the state of the SoftDevice (enabled or disabled).

SVC number allocation	S110 enabled	S110 disabled
Application	0x00-0x0F	0x00-0x0F
SoftDevice	0x10-0xFF	0x10-0xFF

Table 6 *SVC number allocation*

7 Performance

7.1 Interrupt Latency

Latency, additional to ARM® Cortex™-M0 hardware architecture latency, is introduced by SoftDevice logic to manage interrupt events.

Interrupt	Additional latency in CPU cycles
Open peripheral interrupt	50
Blocked or restricted peripheral interrupt (only forwarded when SoftDevice disabled)	63
Application SVC interrupt	14

See ***Table 4 on page 15*** for open, blocked, and restricted peripherals.

7.2 Processor availability

Figure 4 illustrates the parameter values in ***Table 7 on page 18***. The SoftDevice architecture chapter of the nRF51 Reference Manual describes exception (interrupt) management in SoftDevices and is required knowledge for this section.

The parameters of interest are defined around LowerStack and UpperStack exceptions. These exceptions process real time protocol events and API calls (or deferred internal SoftDevice tasks) respectively.

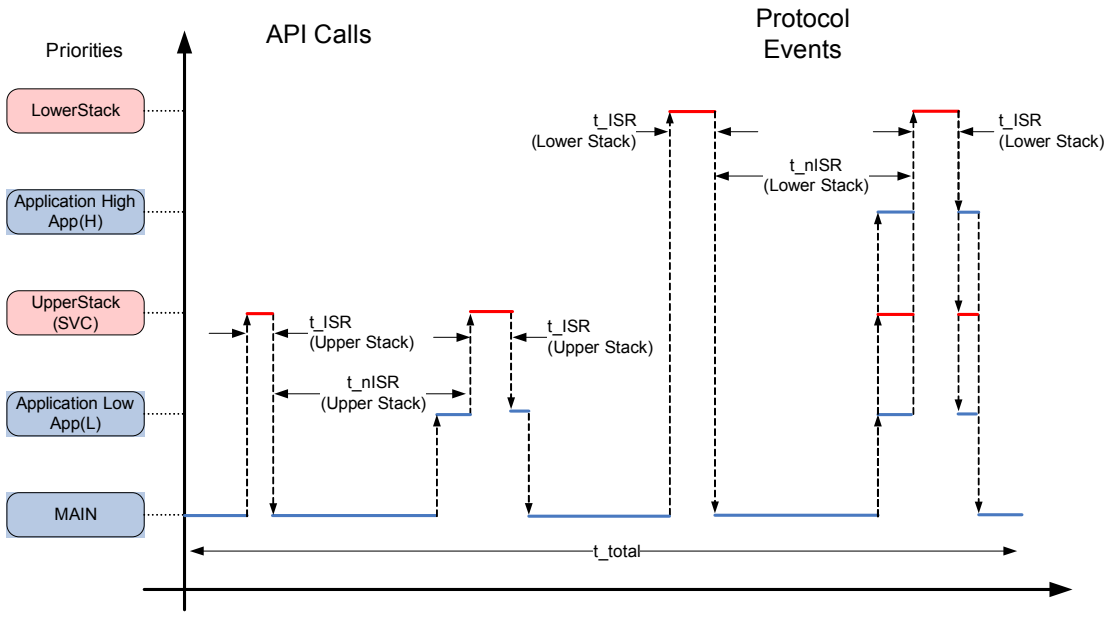


Figure 4 *Interrupt latencies due to SoftDevice processing*

		LowerStack			UpperStack		
Parameter	**Description**	**Min**	**Nom**	**Max**	**Min**	**Nom**	**Max**
t_ISR	Maximum interrupt processing time	-	TBD	400 µs	-	TBD	500 µs
t_nISR	Minimum time between interrupts	230 µs	-	-	[1]	-	-
duty_cycle	Maximum percentage time in interrupt	[2]	2	2	1	1	1

1. This data is application dependent. Calls to the SoftDevice API trigger the Upper Stack interrupt.
2. This data is stack-dependent. See ***Table 8*** for examples of timing based on BLE connection configuration.

Table 7 *S110 interrupt latency numbers*

Application Low, App(L), can be blocked for a maximum of:

$$App(L)_{latency_{\max}} = t_{ISR_{\max(Upper\ Stack)}} + t_{ISR_{\max(Lower\ Stack)}}$$

Application High, App(H), can be blocked for a maximum of:

$$App(H)_{latency\_\max} = t_{ISR_{\max(Lower\ Stack)}}$$

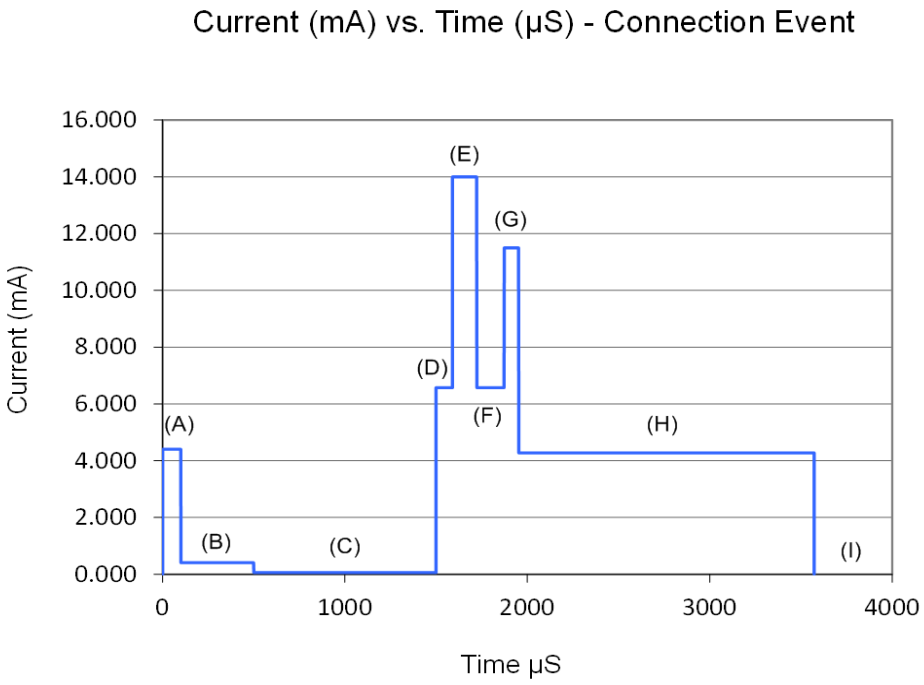
BLE connection configuration	Duty_cycle (Lower Stack)
Connection interval 4s No data transfer	TBD%
Connection interval 100ms 1 packet transfer per event	TBD%
Connection interval 7.5ms 4 packet transfer per event	TBD%

Table 8 *LowerStack interrupt duty cycle examples*

8 Power profiles

This chapter provides power profiles for MCU activity around *Bluetooth* low energy radio events implemented in the S110 SoftDevice. These profiles give a detailed overview of the stages of a radio event, the approximate timing of stages within the event, and how to calculate the peak current at each stage.

8.1 Connection event

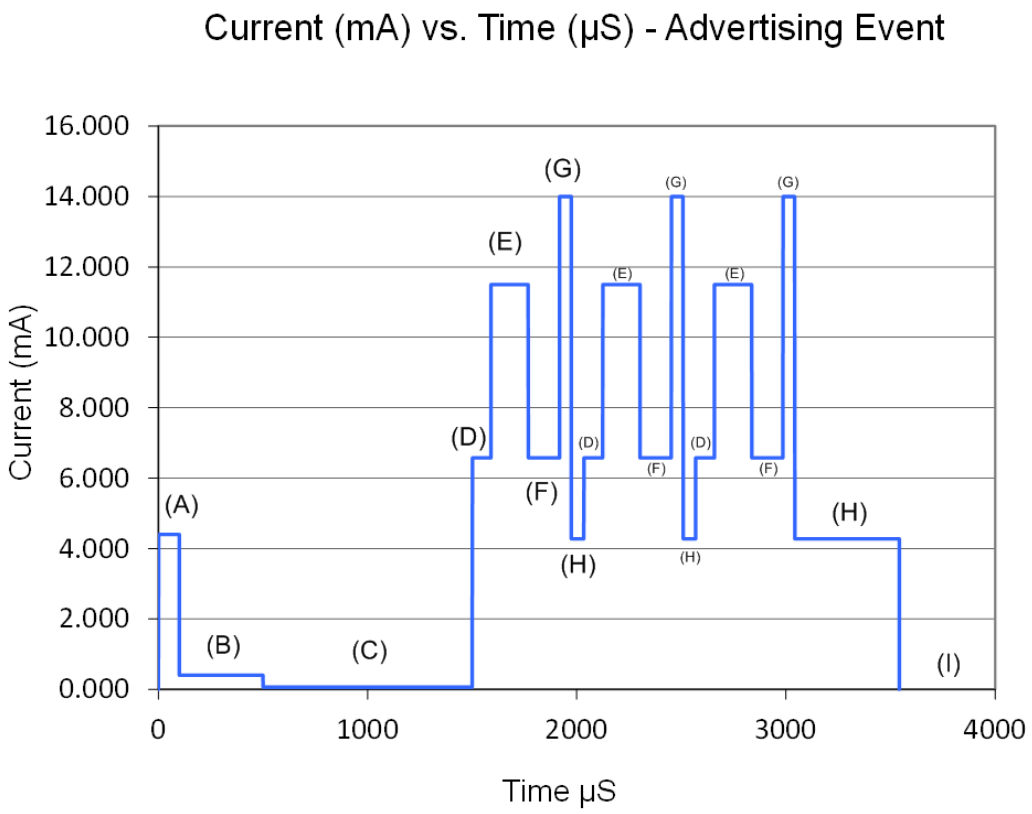


Stage	Description	Current Calculations[1]
(A)	Preprocessing	$I_{ON} + I_{RTC} + I_{X32k} + I_{CPU,Flash}$
(B)	Standby + XO ramp	$I_{ON} + I_{RTC} + I_{X32k} + I_{START,X16M}$
(C)	Standby	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M}$
(D)	Radio Start	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M} + \int (I_{START,RX})$
(E)	Radio RX	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M} + I_{RX}$
(F)	Radio turn-around	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M} + \int (I_{START,TX})$
(G)	Radio TX	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M} + I_{TX,0dBM}$
(H)	Post-processing	$I_{ON} + I_{RTC} + I_{X32k} + I_{CPU,Flash}$
(I)	Idle - connected	$I_{ON} + I_{RTC} + I_{X32k}$

1. See the nRF51822 Product Specification for the symbol values.

Note: I_{RC32k} should be substituted for I_{X32k} when using the 32k RCOSC

8.2 Advertising event



Stage	Description	Current Calculation[1]
(A)	Pre-processing	$I_{ON} + I_{RTC} + I_{X32k} + I_{CPU,Flash}$
(B)	Standby + XO ramp	$I_{ON} + I_{RTC} + I_{X32k} + I_{START,X16M}$
(C)	Standby	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M}$
(D)	Radio start/switch	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M} + \int (I_{START,TX})$
(E)	Radio TX	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M} + I_{TX,0dBM}$
(F)	Radio turn-around	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M} + \int (I_{START,RX})$
(G)	Radio RX	$I_{ON} + I_{RTC} + I_{X32k} + I_{X16M} + I_{RX}$
(H)	Post-processing	$I_{ON} + I_{RTC} + I_{X32k} + I_{CPU,Flash}$
(I)	Idle	$I_{ON} + I_{RTC} + I_{X32k}$

1. See the nRF51822 Product Specification for the symbol values.

Note: I_{RC32k} should be substituted for I_{X32k} when using the 32k RCOSC

9 SoftDevice compatibility and selection

9.1 S110 SoftDevice identification and revision scheme

S110 SoftDevices will be identified by the S110 part code, a qualified IC partcode (for example, nRF51822) and a version number consisting of Major, Minor and Revision numbers.

For example: S110_nRF51822_1.2.3, where major = 1, minor = 2 and, revision = 3.

Table 9 outlines how version numbers are incremented.

Revision	Description
Major increments	Modifications to the API or the function or behaviour of the implementation or part of it have changed. Changes as per Minor Increment may have been made. Application code will not be compatible without some modification.
Minor increments	Additional features and/or API calls are available. Changes as per Revision Increment may have been made. Application code may be modified to take advantage of new features
Revision increments	Issues have been resolved or improvements to performance implemented. Existing application code will not require any modification.

Table 9 *Revision scheme*

Additionally, for revisions of the SoftDevice which are not production qualified, the qualification level, alpha or beta, and sequence number will be appended.

For example:

- An alpha qualified revision:
  S110_nRF51822_1.2.3.alphaX
- The same SoftDevice with beta qualification:
  S110_nRF51822_1.2.3.betaY
- The same SoftDevice with production qualification:
  S110_nRF51822_1.2.3

Table 10 outlines the test qualification levels:

Qualification	Description
Alpha	Development release suitable for prototype application development. Hardware integration testing is not complete. Known issues may not be fixed between alpha releases. Incomplete and subject to change.
Beta	Development release suitable for application development. In addition to alpha qualification: Hardware integration testing is complete. Stable, but may not be feature complete and may contain known issues. Protocol implementations are tested for conformance and interoperability.
Production	Qualified release suitable for product integration. In addition to beta qualification Hardware integration tested over supported range of operating conditions. Stable and complete with no known issues Protocol implementations conform to standards

Table 10 *Test qualification levels*

9.2 Communication of SoftDevice revision updates

When new versions of a SoftDevice become available or the qualification status of a given revision of a SoftDevice is changed, product update notifications will be automatically forwarded, by email, to all users who have a profile configured to receive notifications from the Nordic Semiconductor website.