Proprietary Flex SDK 2.6.1.0 GA
19Q2 Gecko SDK Suite
July 19, 2019

The Proprietary Flex SDK is a complete software development suite for proprietary wireless applications.Per its namesake, Flex offers two implementation options.

The first uses Silicon Labs RAIL (Radio Abstraction Interface Layer), an intuitive and easily-customizable radio interface layer designed to support both proprietary and standards-based wireless protocols.

The second uses Silicon Labs Connect, an IEEE 802.15.4-based networking stack designed for customizable broad-based proprietary wireless networking solutions that require low power consumption and operates in either the sub-GHz or 2.4 GHz frequency bands. The solution is targeted towards simple network topologies.

The Flex SDK is supplied with extensive documentation and sample applications. All examples are provided in source code within the Flex SDK sample applications.

These release notes cover SDK version(s):

2.6.1.0 released July 19, 2019
2.6.0.0 released June 7, 2019

Proprietary

CONNECT APPS AND STACK KEY FEATURES

- Dynamic Multi-Protocol (DMP) BLE + CONNECT
- NVM3 support
- 500 kbps DSSS-OQPSK PHY
- Selective join
- OTA unicast resume

RAIL LIBRARY KEY FEATURES

- Added support for synchronizing the RAIL time base to the PLFRCO on EFR32xG13 Rev D devices
- Improved the configuration switch time in dynamic multiprotocol applications
- Added a new Packet Trace (PTI) message for switching protocols in dynamic multiprotocol applications
- Improved the LQI value returned for IEEE 802.15.4-based PHYs

Compatibility and Use Notices

If you are new to the Silicon Labs Flex SDK, see Using This Release.

Compatible Compilers:

IAR Embedded Workbench for ARM (IAR-EWARM) version 8.30.1

- Using wine to build with the IarBuild.exe command line utility or IAR Embedded Workbench GUI on macOS or Linux could result in incorrect files being used due to collisions in wine's hashing algorithm for generating short file names.
- Customers on macOS or Linux are advised not to build with IAR outside of Simplicity Studio. Customers who do should carefully verify that the correct files are being used.

GCC (The GNU Compiler Collection) version 7.2.1, provided with Simplicity Studio.

Contents

1 General Release Information

The following items apply across Flex SDK components.

1.1 New Items

Added in release 2.6.0.0

Gecko Platform release notes are now available through Simplicity Studio's Launcher Perspective, under **SDK Documentation > Flex SDK 2.6.n.n > Release Notes**. The Gecko Platform code provides functionality that supports Connect protocol plugins and APIs in the form of drivers and other lower layer features that interact directly with Silicon Labs chips and modules. As well as the RAIL Library that is a component of the Flex SDK, Gecko Platform components include EMLIB, EMDRV, NVM3, and mbedTLS.

1.2 Deprecated Items

Deprecated in release 2.6.1.0

Deprecated item: EFR32BG14 Part Support

> **Reason:** The EFR32BG14 is EOL.
>
> **End-of-Service (EoS) Date:** June 30, 2020. As of this EoS date, EFR32BG14 part support will no longer be available in the then current and future GSDK releases, and EFR32BG14 parts will no longer be supported in any GSDK releases.
>
> **Maintenance Period:** From now until the EoS date, only critical bug fixes and security patches may be made available on currently supported GSDK releases.
>
> **Replacement:** EFR32BG13 or EFR32FG14

Deprecated in release 2.6.0.0

As of June 2019 Simplicity Studio 3.0 is being deprecated. All support for Simplicity Studio 3 is scheduled to end December 1st 2019.

2 Applications

2.1 New Items

Added in release 2.6.1.0

RAIL: Range Test BLE and IEEE802.15.4: This application demonstrates over the air range of BRD4171A, BRD4180A and BRD4181A boards. The application has the following predefined PHYs: BLE (125kbps, 500kbps,1Mbps, 2Mbps) and IEEE80215.4 (250kbps).

Added in release 2.6.0.0

Connect (SoC): Empty Example example application: A minimal Connect project structure, used as a starting point for custom applications.

Connect (SoC): Empty Example – DMP example application: A dynamic multiprotocol minimal project structure, used as a starting point for custom applications that run Connect and BLE protocols simultaneously. By default, only the bare minimum plugins and emberAfMainInitCallback are enabled.

Connect (SoC): Demo Connect Light example application: This is a sample application demonstrating a light application that can be turned on/off by a switch application. This application is part of the Connect Light/Switch demo setup.

Connect (SoC): Demo DMP Connect Switch: This is a sample application demonstrating a switch application using dynamic multiprotocol (Connect + BLE). This application is part of the Connect Light/Switch demo setup.

I-grade QFN68 EFR32BG12 part support: part support added for EFR32BG12P232F512IM68-C, EFR32BG12P232F1024IM68-C, EFR32BG12P433F1024IM68-C.

2.2 Improvements

Changed in release 2.6.0.0

OTA unicast: OTA bootloader test plugins updated to demonstrate the "resume" feature.

OTA unicast: Network analyzer decoder updated so that it can decode the new handshake response message correctly.

Connect decoder: Updated the Connect network analyzer decoder for the "extended association request" command so that the newly added payload field gets visualized correctly.

2.3 Fixed Issues

Fixed in release 2.6.1.0

ID #	Description
406623	Fix sample application compatibility of ZG, ZGM and MGM devices (they are now supported only by railtest)
407274	Fix project generation for light and switch sample applications (to resolve the “No toolchain” issue)

Fixed in release 2.6.0.0

ID #	Description
368510	Fixed a railtest issue if receive was entered because of a transmit state transition where we would not change the channel when calling setChannel and would ignore calls to `rx 1` to enter normal receive mode.
394431	Fixed an issue in railtest where the configured antenna diversity settings in HAL config would not be applied to the radio at startup.

2.4 Known Issues in the Current Release

Issues in bold were added since the previous release.

None

2.5 Deprecated Items

None

2.6 Removed Items

None

3 Connect Stack

3.1 New Items

3.1.1 Application Framework API

Added in release 2.6.0.0

- Added the ability for a OTA unicast bootloader client to resume an image download process at any point. This is accomplished by passing a "start index" to the emberAfPluginOtaUnicastBootloaderClientNewIncomingImageCallback() callback.

  To this purpose, the emberAfPluginOtaUnicastBootloaderClientNewIncomingImageCallback() callback has been modified, it now passes two extra parameters: an imageSize integer indicating the total size in bytes of a new incoming image. A startIndex integer pointer which can be used to change the index of the first byte to be downloaded during the image downloading process. This value is set by default to 0 (that is, by default the client starts at the beginning of the image).
- Added a new plugin option to the mailbox client plugin that allows the user to configure the handshake timeout. A node running as star topology sleepy end device should configure this parameter consistently with the short poll plugin option in the poll plugin so that the end device polls quickly enough to get the handshake response from the mailbox server before timing out.
- A new “mbed Tls” plugin has been added. When this plugin is enabled all security operations are performed through the mbed Tls code.
- Added a new “NVM3” plugin. When this plugin is enabled Silicon Labs NVM3 non-volatile memory storage system will be used. The NVM3 provides a means to store and retrieve objects (key/value pair) from the flash and provides wear leveling to reduce erase and write cycles to maximise the lifetime of the flash pages. Objects in NVM3 can either be accessed directly through the native NVM3 API or through the token API in the same way as SimEE1 or SimEE2 based tokens. This library requires the Simulated EEPROM version 2 to NVM3 upgrade library or stub upgrade library. The number of flash pages to use for the NVM3 storage is configurable through the plugin options. IMPORTANT: When compiling for a device which already contains NVM3 data, the number of flash pages configured for the compilation must match the number of flash pages used for the existing NVM3 instance on the device.

3.1.2 Stack API

For additional documentation please refer to the Connect API Reference Guide.

Added in release 2.6.0.0

- Introduced Dynamic Multiprotocol support Connect + BLE. The BLE and the Connect stacks can now be run concurrently on the same SoC. Please refer to DMP_DOCUMENT_REFERENCE for more information.
- Added support for selective joining. If the selective joining payload is configured at a star coordinator or a star range extender node, that node will allow to the network only joining nodes including the same joining payload in the Connect Extended Association Request command. Both joiner and joinee nodes can configure such payload using the new API emberSetSelectiveJoinPayload(). A second API emberClearSelectiveJoinPayload() is oprovided to clear the joining payload.
- Added support for securing short source addressed frames when running in MAC mode. The receiver side in order to properly decrypt such messages is required to populate a short-to-long mapping. The long ID of the source node is required in order to construct the security nonce. This can be accomplished by invoking the new API emberMacAddShortToLongAddressMapping(). A second API emberMacClearShortToLongAddressMappings() is provided to clear the entire table.
- The parent support stack library now has a new option "First Assigned Short ID". In Connect, short IDs are assigned by the star coordinator node sequentially. This option allows to specify at a star coordinator node the beginning of the assigned IDs interval. In so doing, a pool of addresses can be reserved for other use (e.g., for star end device nodes that request a specific short ID rather than having the coordinator assigning one).

3.2 Improvements

3.2.1 Application Framework API

Changed in release 2.6.0.0

- The **emberAfPluginOtaUnicastBootloaderClientAbortImageDownload()** API in the OTA Unicast Bootloader plugin has been changed: the “applicationStatus” parameter has been removed.

- The **emberAfPluginOtaUnicastBootloaderClientIncomingRequestBootloadCallback()** callback in the OTA Unicast Bootloader plugin has been changed: the “applicationStatus” parameter has been removed.

3.2.2 Stack API

Changed in release 2.6.0.0

- A new “ackRssi” field has been added to the **EmberIncomingMessage** and **EmberIncomingMacMessage** structs. This new field provides the RSSI of the received ACK (if any) corresponding to the incoming message.

3.3 Fixed Issues

Fixed in release 2.6.1.0

ID #	Description
	Fixed an issue whereas using the **emberMacSetParams()** to disable CCA would result in an assert at the MAC layer.

Fixed in release 2.6.0.0

ID #	Description
	The **emberSetRadioPower()** API now saves the passed value in persistent storage if the passed TX power is a legal value. The saved value is restored after reboot together with the rest of the saved network parameters when the **emberNetworkInit()** API is called.

3.4 Known Issues in the Current Release

Issues in bold were added since the previous release.

- When running the RAIL Multiprotocol Library (used for example when running DMP Connect+BLE), IR Calibration is not performed because of a know issue in the RAIL Multiprotocol Library. As result, there is an RX sensitivity loss in the order of 3 or 4 dBm.

3.5 Deprecated Items

None

3.6 Removed Items

3.6.1 Stack API

Removed in release 2.6.0.0

- The deprecated **emberMacSetAllocateAddressFlag()** stack API has now been removed.

4 RAIL Library

For additional documentation please refer to the RAIL API Reference Guide.

4.1 New Items

Added in release 2.6.1.0

Added RAIL_TX_POWER_LEVEL_MAX which can be used to set the max PA power level across PA's. RAIL_TX_POWER_LEVEL_INVALID was added in 2.7.0 as the value 255. Some customers were using 255 to set max power across PA's with \ref RAIL_SetTxPower, which previously worked, but will now return an error.

Added in release 2.6.0.0

Added new 802.15.4 RAIL APIs – RAIL_IEEE802154_EnableEarlyFramePending() and RAIL_IEEE802154_EnableDataFramePending() – to support Thread 1.2 enhanced frame pending feature.

Added new 802.15.4 RAIL APIs – RAIL_IEEE802154_ConfigGOptions() and RAIL_IEEE802154_ConfigEOptions() – for configuring certain 802.15.4E-2012 and G-2012 features needed by GB868.

Added support for Z-Wave node ID based packet filtering via the RAIL_ZWAVE_OPTION_NODE_ID_FILTERING option.

Added support to RAIL_Sleep() for the PLFRCO on EFR32xG13 Rev D parts.

Added information to packet trace for every protocol switch in dynamic multiprotocol, telling the user what protocol we have changed to, as well the radio event that triggered this switch. This information is visible in Network Analyzer for better debugging of DMP applications.

Added support for the radio sending an ACK packet automatically when in Z-Wave mode as long as node ID filtering and Auto_Ack features are enabled and a packet requesting an ACK is sent to the device.

Added a new API – RAIL_UseDma() – which can be used to enhance RAIL startup speed, if called before RAIL_Init().

4.2 Improvements

Changed in release 2.6.0.0

Reduced switch time overhead for dynamic multiprotocol applications. The new switch times as well as information about them are documented in the rail_multiprotocol page.

Changed the LQI metric for the 2.4GHz IEEE802.15.4 PHY configurations to be scaled from 0 - 255 and to include more data to make it more stable. This can impact existing applications that are using the LQI values returned in prior RAIL versions.

In RAILTest, if receive was entered because of a transmit state transition, we would not change the channel when calling setChannel and would ignore calls to `rx 1` to enter normal receive mode.

Improved documentation of RAIL_RxPacketStatus_t values and their corresponding RAIL_Events_t events.

Use of the unsafe enum GPIO_Port_TypeDef within RAIL aggregate types RAIL_PtiConfig_t and RAIL_AntennaConfig_t has been replaced by safe uint8_t.

Clarified that RAIL_EVENT_TX_BLOCKED and RAIL_EVENT_TX_CHANNEL_BUSY leave the TX FIFO intact without consuming any of its packet data.

4.3 Fixed Issues

Fixed in release 2.6.1.0

ID	Description
337468	Fixed an issue where calling RAIL_Sleep() when no RAIL events are pending would not stop and synchronize the clock source as requested, but instead it would return success indicating that it had. The clock source will now be properly stopped and synchronized even if there are no events pending, and it will be the user's responsibility to wake up on time.
337468	Fixed an issue where calling RAIL_Wake() without first successful calling RAIL_Sleep() could cause the clock source to drift even when using RAIL_SLEEP_CONFIG_TIMERSYNC_ENABLED.
389462	Fixed an issue with RAIL_Calibrate() in multiprotocol, which would return RAIL_STATUS_INVALID_STATE if it is called with an inactive railHandle. Now, RAIL_Calibrate() will make the given railHandle active, if not already, and perform calibration.
389462	Fixed an issue with RAIL_Calibrate() where after completing calibration the radio would no longer be able to receive any packets, sometimes, until it was reset.

401826	Fixed several issues and race conditions with RAIL_ScheduleRx() window-end handling and event reporting to conform to its intended design:
	• When the RX window ends with no RAIL_EVENTS_RX_COMPLETION imminent, RAIL_EVENT_RX_SCHEDULED_RX_END is posted.
	• When the RX window is implicitly ended by one of the RAIL_EVENTS_RX_COMPLETION – for example, because it results in an RX transition to Idle or because RAIL_ScheduleRxConfig_t::rxTransitionEndSchedule is non-zero – the event(s) posted depend on that setting:
	  ○ When zero, both events are posted simultaneously.
	  ○ When non-zero, only the appropriate RAIL_EVENTS_RX_COMPLETION is posted, unless that event is not enabled, in which case RAIL_EVENT_RX_SCHEDULED_RX_END is substituted instead.
	• When the RX window ends while receiving a packet, it is deferred to the expected RX completion event (which includes aborting that packet when RAIL_ScheduleRxConfig_t::hardWindowEnd is non-zero). Event(s) reported at that time are the same as in the previous case.

406276	Restored ability for RAIL_StartCcaCsmaTx() and RAIL_StartCcaLbtTx() to perform an immediate transmit when their respective RAIL_CsmaConfig_t::csmaTries or RAIL_LbtConfig_t::lbtTries is 0.This functionality was improperly removed in 2.6.0.
406302	Resolved multiple compilation issues related to building with HAL_COEX_RUNTIME_PHY_SELECT and HAL_ANTDIV_RX_RUNTIME_PHY_SELECT macros enabled.
407047	Fixed an EFR32xG21 issue where CSMA/LBT CCA durations were significantly shorter than specified.
408096	Fixed the 802.15.4 ACK turnaround time on the EFR32xG21 platform. Due to a calculation error this was actually 18us too short which could cause interoperability problems.

Fixed in release 2.6.0.0

ID	Description
360371	Fixed an issue where calling RAIL_GetTxPowerDbm prior to calling RAIL_SetTxPower would return -500 (i.e., -50dBm). As a part of the fix, we now return an invalid dBm value, RAIL_TX_POWER_MIN, if RAIL_SetTxPower was not called before calling RAIL_GetTxPowerDbm or if RAIL_SetTxPower it returns an error status.
370805	Fixed an issue with the EFR32xG21 reporting a phantom packet on PTI after reset.
376229	Fixed an issue with Rx antenna diversity operation that prevented CCA from working, causing CSMA failures.
392350	Corrected an issue where the radio might be left in receive after a RAIL_EVENT_TX_BLOCKED or RAIL_EVENT_TX_CHANNEL_BUSY when the transmit RAIL_StateTransitions_t::error is RAIL_RF_STATE_IDLE.
400303	Corrected an issue where RAIL_EVENT_TX_CHANNEL_BUSY due to RAIL_CsmaConfig_t::csmaTimeout or RAIL_LbtConfig_t::lbtTimeout would prevent further transmits.
400303	Corrected an issue where an invalid RAIL_CsmaConfig_t::ccaDuration or RAIL_LbtConfig_t::lbtDuration too large for the radio configuration to handle would not fail the respective transmit; this now returns RAIL_STATUS_INVALID_PARAMETER.

4.4 Known Issues in the Current Release

Issues in bold were added since the previous release.

None

4.5 Deprecated Items

Deprecated in release 2.6.0.0

Use of the RAIL_BLE_Coding_t values RAIL_BLE_Coding_125kbps_DSA and RAIL_BLE_Coding_500kbps_DSA is deprecated. These should be replaced with the more generic RAIL_BLE_Coding_125kbps and RAIL_BLE_Coding_500kbps values respectively. For now, choosing either value will result in the same underlying behavior.

4.6 Removed Items

None

5 Using This Release

This release contains the following

- Radio Abstraction Interface Layer (RAIL) stack library
- Connect Stack Library
- RAIL and Connect Sample Applications
- RAIL and Connect Plugins and Application Framework

For more information about the Flex SDK see UG103.13: RAIL Fundamentals and UG103.12: Silicon Labs Connect Fundamentals. If you are a first time user, see QSG138: Getting Started with the Silicon Labs Flex Software Development Kit for the Wireless Gecko (EFR32) Portfolio.

5.1 Installation and Use

Stack installation instruction are covered in QSG138: Getting Started with the Silicon Labs Flex Software Development Kit for the Wireless Gecko (EFR32) Portfolio.

Use the Flex SDK with the Silicon Labs Simplicity Studio V4 development platform. Simplicity Studio ensures that most software and tool compatibilities are managed correctly. Install software and board firmware updates promptly when you are notified.

Documentation specific to the SDK version is installed with the SDK. Additional information can often be found in the knowledge base articles (KBAs). API references and other information about this and earlier releases is available on https://docs.silabs.com/.

5.2 Support

Development Kit customers are eligible for training and technical support. You can use the Silicon Laboratories Flex web page to obtain information about all Silicon Labs Thread products and services, and to sign up for product support.

You can contact Silicon Laboratories support at http://www.silabs.com/support.

6 Legal

6.1 Disclaimer

Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications.

Application examples described herein are for illustrative purposes only.

Silicon Labs reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Labs shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any Life Support System. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

6.2 Trademark Information

Silicon Laboratories Inc.® , Silicon Laboratories®, Silicon Labs®, SiLabs® and the Silicon Labs logo®, Bluegiga®, Bluegiga Logo®, Clockbuilder®, CMEMS®, DSPLL®, EFM®, EFM32®, EFR, Ember®, Energy Micro, Energy Micro logo and combinations thereof, "the world’s most energy friendly microcontrollers", Ember®, EZLink®, EZRadio®, EZRadioPRO®, Gecko®, ISOmodem®, Micrium, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress®, Zentri, Z-Wave and others are trademarks or registered trademarks of Silicon Labs.

ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings.

Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.