Proprietary Flex SDK 3.7.3.0 GA
Gecko SDK Suite 4.4
May 2, 2024

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):

3.7.3.0 GA released May 2, 2024.
3.7.2.0 GA released April 10, 2024.
3.7.1.0 GA released February 14, 2024.
3.7.0.0 GA released December 13, 2023.

Compatibility and Use Notices

For information about security updates and notices, see the Security chapter of the Gecko Platform Release notes installed with this SDK or on the TECH DOCS tab on https://www.silabs.com/developers/flex-sdk-connect-networking-stack. Silicon Labs also strongly recommends that you subscribe to Security Advisories for up-to-date information. For instructions, or if you are new to the Silicon Labs Flex SDK, see Using This Release.

Compatible Compilers:

IAR Embedded Workbench for ARM (IAR-EWARM) version 9.40.1

- 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 12.2.1, provided with Simplicity Studio.
1 Connect Applications

This release of the Gecko SDK (GSDK) will be the last with combined support for all EFM and EFR devices, except for patches to this version as needed. Starting in mid-2024 we will introduce separate SDKs:

- The existing Gecko SDK will continue with support for Series 0 and 1 devices.
- A new SDK will cater specifically to Series 2 and 3 devices.

The Gecko SDK will continue to support all Series 0 and 1 devices with no change to the long-term support, maintenance, quality, and responsiveness provided under our software policy.

The new SDK will branch from Gecko SDK and begin to offer new features that help developers take advantage of the advanced capabilities of our Series 2 and 3 products.

This decision aligns with customer feedback, reflecting our commitment to elevate quality, ensure stability, and enhance performance for an exceptional user experience across our software SDKs.

1.1 New Items

**Added in release 3.7.0.0**

- EFR32xG28 Explorer Kit Support
- EFRBG22-E Support
- MG24 QFN40 Support

1.2 Improvements

**Changed in release 3.7.1.0**

- Connect NCP: Installed the bootloader interface component by default. It allows to perform firmware updates from the host.

**Changed in release 3.7.0.0**

- Gecko Bootloader is used instead of Legacy HAL

1.3 Fixed Issues

**Fixed in release 3.7.0.0**

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
<th>Workaround</th>
</tr>
</thead>
<tbody>
<tr>
<td>1076409</td>
<td>OTA Bootloader is not working on Series2</td>
<td></td>
</tr>
</tbody>
</table>

1.4 Known Issues in the Current Release

Issues in bold were added since the previous release. If you have missed a release, recent release notes are available on the TECH DOCS tab on [https://www.silabs.com/developers/flex-sdk-connect-networking-stack](https://www.silabs.com/developers/flex-sdk-connect-networking-stack).

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
<th>Workaround</th>
</tr>
</thead>
<tbody>
<tr>
<td>652925</td>
<td>EFR32XG21 is not supported for &quot;Flex (Connect) - SoC Light Example DMP&quot; and &quot;Flex (Connect) - SoC Switch Example&quot;</td>
<td></td>
</tr>
<tr>
<td>1139850</td>
<td>DMP instabilities with XG27</td>
<td></td>
</tr>
</tbody>
</table>
1.5 Deprecated Items

Deprecated in release 3.7.0.0

None

1.6 Removed Items

Removed in release 3.7.0.0

None
2 Connect Stack

2.1 New Items

Added in release 3.7.0.0

- Added support of SUN-FSK and SUN-OFDM. Enable the transmission and reception of packets carrying up to 2033 bytes of payload. `EmberMessageLength` was extended to allow the message length to be coded on 16 bits. Most of the packet's buffer are allocated dynamically except the radio RX fifo that is static and has to be extended to be able to contain at least one maximum length packet. It is the application responsibility to do so by implementing `RAILCb_SetupRxFifo()` and calling `RAIL_SetRxFifo()`. Depending on the application configuration, RTOS task stack sizes, minimum heap size and CPC packet maximum length may also need to be increased.
- Added new API `emberOfdmSetMcs()` and `emberOfdmGetMcs()` that set and get the OFDM modulation and coding scheme (MCS).
- Updated the hardware abstraction layer.

2.2 Improvements

Changed in release 3.7.1.0

- Reduced Connect Serial Protocol RAM usage

Changed in release 3.7.0.0

None

2.3 Fixed Issues

Fixed in release 3.7.2.0

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1252147</td>
<td>Fixed a new usage of the assert function that caused a significant increase of our flash usage. The problem fixed came from the fact that the value of the <code>expression</code> parameter is saved in a string format. v3.7.0 made use of the standard assert function while the previous versions used an implementation from the legacy hardware abstraction layer. The <code>expression</code> parameter was then always forced to 0, and the assertion test was made prior to the function call. V3.7.2 reverts that change.</td>
</tr>
<tr>
<td>1121468</td>
<td>Fixed an issue that was causing RAIL Utility PA configuration to be ignored. It was hard-coded in the libraries.</td>
</tr>
<tr>
<td>1266682</td>
<td>Fixed an issue that was causing the range extender update period configuration to be ignored. It was hard-coded in the libraries.</td>
</tr>
</tbody>
</table>

Fixed in release 3.7.1.0

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1223893</td>
<td>Fixed Connect SUN PHY signaling in the radio configurator artifacts. The lack of a good signalization was preventing the lower MAC to detect which PHY profile was used.</td>
</tr>
</tbody>
</table>
Fixed in release 3.7.0.0

None

2.4 Known Issues in the Current Release

Issues in bold were added since the previous release. If you have missed a release, recent release notes are available on the TECH DOCS tab on https://www.silabs.com/developers/gecko-software-development-kit.

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
<th>Workaround</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>When running the RAIL Multiprotocol Library (used for example when running DMP Connect+BLE), IR Calibration is not performed because of a known issue in the RAIL Multiprotocol Library. As result, there is an RX sensitivity loss in the order of 3 or 4 dBm.</td>
<td>Until this is changed to properly pull from the configuration header, the file ember-phy.c in the user's project will need to be modified by hand to reflect the desired PA mode, voltage, and ramp time.</td>
</tr>
<tr>
<td>501561</td>
<td>In the Legacy HAL component, the PA configuration is hard-coded regardless of the user or board settings.</td>
<td></td>
</tr>
<tr>
<td>711804</td>
<td>Connecting multiple devices simultaneously may fail with a timeout error.</td>
<td></td>
</tr>
</tbody>
</table>

2.5 Deprecated Items

**Deprecated in release 3.7.0.0**

None

2.6 Removed Items

**Removed in release 3.7.0.0**

None
3 RAIL Applications

This release of the Gecko SDK (GSDK) will be the last with combined support for all EFM and EFR devices, except for patches to this version as needed. Starting in mid-2024 we will introduce separate SDKs:

- The existing Gecko SDK will continue with support for Series 0 and 1 devices.
- A new SDK will cater specifically to Series 2 and 3 devices.

The Gecko SDK will continue to support all Series 0 and 1 devices with no change to the long-term support, maintenance, quality, and responsiveness provided under our software policy.

The new SDK will branch from Gecko SDK and begin to offer new features that help developers take advantage of the advanced capabilities of our Series 2 and 3 products.

This decision aligns with customer feedback, reflecting our commitment to elevate quality, ensure stability, and enhance performance for an exceptional user experience across our software SDKs.

3.1 New Items

**Added in release 3.7.0.0**

- EFR32xG28 Explorer Kit Support
- EFRBG22-E Support
- MG24 QFN40 Support

3.2 Improvements

**Changed in release 3.7.0.0**

- EFR32xG28 Proprietary 2.4 GHz 15.4 Standard PHY Support
- Connect OFDM PHY Support for
  - RAIL - SoC Simple TRX
  - RAIL - SoC Range Test

3.3 Fixed Issues

**Fixed in release 3.7.0.0**

None

3.4 Known Issues in the Current Release

None

3.5 Deprecated Items

**Deprecated in release 3.7.0.0**

None

3.6 Removed Items

**Removed in release 3.7.0.0**

None
4 RAIL Library

4.1 New Items

Added in release 3.7.2.0

None

Added in release 3.7.1.0

- Added support for setting the Whitening and CRC initial values at runtime that are defined by the existing PHY's radio configuration.

Added in release 3.7.0.0

- Added support for a new assert, which will be thrown if the loaded PHY is not supported by the software-defined modem on the EFR32xG25.
- Added new RAIL_GetAutoAckFifo() API and allow NULL for RAIL_WriteAutoAckFifo() or RAIL_IEEE802154_WriteEnhAck() ackData parameter, which gives applications direct access to the AutoAck FIFO to construct Ack packets in pieces.
- Added support for antenna selection through the applicable RAIL_RxOptions_t and RAIL_TxOptions_t values when using OFDM on the EFR32xG25.
- Added a new "RAIL Utility, SFM Sequencer Image Selection" component to allow selection of modulations supported by EFR32xG25 software modem (SFM). These changes can save considerable flash space by reducing the set of modulations to just those that are needed.
- Added support for Sidewalk PHYs on the EFR32xG23 and EFR32xG28 chips.
- Added an assert RAIL_ASSERT_INVALID_XTAL_FREQUENCY on the EFR32xG1x and EFR32xG2x chips, to fire when a radio config that is loaded is incompatible with a device due to the defined crystal frequency of the config not matching with the device's crystal frequency.
- Added the RAIL_TX_REPEAT_OPTION_START_TO_START option to measure the delay between repeated transmits from the start of TX to start of TX instead of the default from end of TX to start of TX.
- Added support for GCC 12.2.1 and IAR 9.40.1 compilers.
- Added support for Fast Channel Switching PHYs on the EFR32xG24.
- Added support for RAIL_IEEE802154_SUPPORTS_G_MODESWITCH on the EFR32xG28.
- Added support for the IEE802154 2.4 GHz coherent PHYs via RAIL_SUPPORTS_IEEE802154_BAND_2P4 on the EFR32xG28.
- Added a new RAIL_RxOptions_t option to enable collision detection on the EFR32xG25. Once enabled, when a collision with a strong enough packet is detected, the demod will stop the current packet decoding and try to detect the preamble of the incoming packet.
- Added support of RAILTEST for the MGM240x modules.
- Added support for channel masks during Wi-SUN mode switch in RAILTEST application on the EFR32xG25.
- Added support of 802.15.4 IMM-ACK with OFDM and OQPSK modulations (FCS 4 bytes only) on the EFR32xG25.
- Updated the RAIL Library component to automatically include the "RAIL Utility, Built-in PHYs Across HFXO Frequencies" component to better support different HFXO frequencies on supported parts.

4.2 Improvements

Changed in release 3.7.2.0

None
**Changed in release 3.7.1.0**

None

**Changed in release 3.7.0.0**

- Updated the default PTI rate to 3200000 bps on the EFR32XG25.
- No longer include `rail_chip_specific.h` and `rail_features.h` (anything dependent on `em_device.h`) when `SLI_LIBRARY_BUILD` is defined. This will allow the user to build their radio code in a way that depends on RAIL generically but is not chip-specific. When doing this the code cannot depend on things that are inherently chip-specific and are still in those files like `RAIL_RF_PATHS`, `RAIL_NUM_PA`, or any of the compile-time `RAIL_SUPPORTS_xxx` defines. Code will need to call appropriate runtime APIs or split itself between the generic and chip specific portions and build them separately.
- To better support chip-agnostic builds `RAIL_TxPowerMode_t` is now a superset representing all possible PAs across all chips. Any code depending on the number of or consecutive ordering of chip-supported PAs will likely need to be updated.
- To better support chip-agnostic builds `RAIL_CalValues_t` and subordinate `RAIL_IrCalValues_t` have grown to encompass the superset of fields needed on all chips, affecting all chips except EFR32xG25.
- To better support chip-agnostic builds `RAIL_TransitionTime_t` and therefore `RAIL_StateTiming_t` have grown to a superset type, affecting EFR32xG1.
- To better support chip-agnostic builds `RAIL_FIFO_ALIGNMENT` is now universally 32-bit, affecting EFR32xG1x and EFR32xG21, but is still only enforced on chips actually requiring that alignment.
- Updated the default power curves for the 10dBm High Power PA on EFR32xG24.

**4.3 Fixed Issues**

**Fixed in release 3.7.2.0**

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1248013</td>
<td>Fixed an issue with IEEE 802.15.4 High Data Rate (HDR) packets such that PTI now properly indicates they have a 2-byte 802.15.4 PHY header.</td>
</tr>
<tr>
<td>1255347</td>
<td>Fixed an issue with <code>RAIL_SetTxFifo()</code> setup that could trip <code>RAIL_ASSERT_FAILED_UNEXPECTED_STATE_TX_FIFO</code> when the prior TX FIFO was larger and contained more data than the new, smaller TX FIFO can hold. Any data in the old TX FIFO should have been ignored.</td>
</tr>
<tr>
<td>1271435</td>
<td>Fixed an issue where <code>RAIL_WriteTxFifo()</code> could write outside the TX FIFO when called without resetting the FIFO in dynamic multiprotocol applications.</td>
</tr>
</tbody>
</table>

**Fixed in release 3.7.1.0**

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1241800</td>
<td>Added the missing <code>pa_dbm_mapping_table.py</code> script to the release package. This script is used to help create power tables on the EFR32xG25 part.</td>
</tr>
<tr>
<td>1242723</td>
<td>Fixed an issue to exit from the critical section while performing multiprotocol PA operations on the EFR32xG25.</td>
</tr>
<tr>
<td>1243727</td>
<td>Improved the CCA fiability on the EFR32xg23, EFR32xg25 and EFR32xg28 chips.</td>
</tr>
</tbody>
</table>
Fixed in release 3.7.0.0

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1079816</td>
<td>Fixed a race condition on the EFR32xG22 and later during RX channel hopping or duty-cycling where frame detection occurring close to when a hop should happen could leave the radio stuck in reception but not receiving anything, with the only remedy being to idle the radio.</td>
</tr>
<tr>
<td>1088439</td>
<td>Fixed an issue which would cause the incorrect antenna to be reported for a received packet on the EFR32xG25 when using OFDM and antenna diversity.</td>
</tr>
<tr>
<td>1153679</td>
<td>Fixed an issue in &quot;RAIL Utility, Coexistence&quot; component on the EFR32xG24 where a GRANT signal pulse less than 100us might result in the radio not being properly placed in hold off after GRANT is deasserted.</td>
</tr>
<tr>
<td>1156980</td>
<td>Fixed an issue with channel hopping on the EFR32xG22 and later where use of RAIL_RX_CHANNEL_HOPPING_OPTION_RSSI_THRESHOLD can prevent the timed RX channel hopping modes including RX duty-cycling from timing out properly.</td>
</tr>
<tr>
<td>1175684</td>
<td>Fixed an issue with the RAIL_IDLE form of RAIL_Idle() and RAIL_STOP_MODE_PENDING form RAIL_StopTx() during the LBT/CSMA phase of a transmit which could previously hang. Pending LBT/CSMA and scheduled transmits are now stopped or idled with RAIL_EVENT_TX_BLOCKED triggered, except for idle mode RAIL_IDLE_FORCE_SHUTDOWN_CLEAR_FLAGS.</td>
</tr>
<tr>
<td>1183040</td>
<td>Updated all PHYs built into RAIL on Series 2 platforms using the latest radio calculator to reduce observed high reference spurs.</td>
</tr>
<tr>
<td>1184982</td>
<td>Fixed an issue with RAIL_StartAverageRssi() that could cause execution to hang in interrupt context just before raising RAIL_EVENT_RSSI_AVERAGE_DONE. This was primarily an issue on the EFR32xG21 platform. Note that it is possible, though unlikely, that RAIL_GetAverageRssi() might still return RAIL_RSSI_INVALID after the average RSSI period has finished.</td>
</tr>
<tr>
<td>1184982</td>
<td>Fixed an issue with RAIL_StartAverageRssi() that caused it to mistakenly idle the radio in the newly-activated protocol if a dynamic protocol switch occurred during the averaging operation of the suspended protocol.</td>
</tr>
<tr>
<td>1188083</td>
<td>Fixed an issue where RAIL_Idle() could hang waiting for the radio to idle when using IEEE 802.15.4 fast RX channel switching.</td>
</tr>
<tr>
<td>1190187</td>
<td>Fixed an issue where idling the radio before a Scheduled Receive window ends could cause a subsequent packet that should be silently filtered and rolled back to instead be received with RAIL_RX_PACKET_READY_CRC_ERROR.</td>
</tr>
<tr>
<td>1201506</td>
<td>Fixed an issue in multiprotocol applications where the wrong sync word would be used if two protocols used the same radio configuration and only one of those protocols set a custom sync word with RAIL_ConfigSyncWords() API.</td>
</tr>
</tbody>
</table>

4.4 Known Issues in the Current Release

Issues in bold were added since the previous release.

<table>
<thead>
<tr>
<th>ID #</th>
<th>Description</th>
<th>Workaround</th>
</tr>
</thead>
<tbody>
<tr>
<td>641705</td>
<td>Using direct mode (or IQ) functionality on EFR32xG23 requires a specifically set radio configuration that is not yet supported by the radio configurator. For these requirements, reach out to technical support who could provide that configuration based on your specification</td>
<td></td>
</tr>
<tr>
<td>732659</td>
<td>Infinite receive operations where the frame's fixed length is set to 0 are not working correctly on the EFR32xG23 series chips.</td>
<td></td>
</tr>
</tbody>
</table>
| 732659  | On EFR32xG23: Wi-SUN FSK mode 1a exhibits a PER floor with frequency offsets around ± 8 to 10 KHz  
Wi-SUN FSK mode 1b exhibits a PER floor with frequency offsets around ± 18 to 20 KHz |            |
4.5 Deprecated Items

Note:

- The RAIL 2.x API is planned for deprecation in the 24Q4-GA release (December 2024). At that time, the new RAIL 3.0 API will be released for all supported chips along with a RAIL 2.x compatibility layer and migration guide.

- The goal of this new API is to evolve what we currently have to get rid of some unused features, add better support for concurrent listening use cases, and simplify channel and PA configurations.

- We hope the migration is straightforward and simple for the majority of customers, but there may be manual help needed in some cases which we will do our best to document to ease this transition.

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 Components and Application Framework

This SDK depends on Gecko Platform. The Gecko Platform code provides functionality that supports protocol plugins and APIs in the form of drivers and other lower layer features that interact directly with Silicon Labs chips and modules. Gecko Platform components include EMLIB, EMDRV, RAIL Library, NVM3, and mbedtls. Gecko Platform release notes are available through Simplicity Studio’s Documentation tab.

For more information about the Flex SDK v3.x see UG103.13: RAIL Fundamentals and UG103.12: Silicon Labs Connect Fundamentals. If you are a first time user, see QSG168: Proprietary Flex SDK v3.x Quick Start Guide.

5.1 Installation and Use

The Proprietary Flex SDK is provided as part of the Gecko SDK (GSDK), the suite of Silicon Labs SDKs. To quickly get started with the GSDK, install Simplicity Studio 5, which will set up your development environment and walk you through GSDK installation. Simplicity Studio 5 includes everything needed for IoT product development with Silicon Labs devices, including a resource and project launcher, software configuration tools, full IDE with GNU toolchain, and analysis tools. Installation instructions are provided in the online Simplicity Studio 5 User’s Guide.

Alternatively, Gecko SDK may be installed manually by downloading or cloning the latest from GitHub. See https://github.com/SiliconLabs/gecko_sdk for more information.

Simplicity Studio installs the GSDK by default in:

- (Windows): C:\Users\<NAME>\SimplicityStudio\SDKs\gecko_sdk
- (MacOS): /Users/<NAME>/SimplicityStudio/SDKs/gecko_sdk

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 Security Information

Secure Vault Integration

When deployed to Secure Vault High devices, sensitive keys are protected using the Secure Vault Key Management functionality. The following table shows the protected keys and their storage protection characteristics.

<table>
<thead>
<tr>
<th>Wrapped Key</th>
<th>Exportable / Non-Exportable</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thread Master Key</td>
<td>Exportable</td>
<td>Must be exportable to form the TLVs</td>
</tr>
<tr>
<td>PSKc</td>
<td>Exportable</td>
<td>Must be exportable to form the TLVs</td>
</tr>
<tr>
<td>Key Encryption Key</td>
<td>Exportable</td>
<td>Must be exportable to form the TLVs</td>
</tr>
<tr>
<td>MLE Key</td>
<td>Non-Exportable</td>
<td></td>
</tr>
<tr>
<td>Temporary MLE Key</td>
<td>Non-Exportable</td>
<td></td>
</tr>
<tr>
<td>MAC Previous Key</td>
<td>Non-Exportable</td>
<td></td>
</tr>
<tr>
<td>MAC Current Key</td>
<td>Non-Exportable</td>
<td></td>
</tr>
<tr>
<td>MAC Next Key</td>
<td>Non-Exportable</td>
<td></td>
</tr>
</tbody>
</table>

Wrapped keys that are marked as “Non-Exportable” can be used but cannot be viewed or shared at runtime.

Wrapped keys that are marked as “Exportable” can be used or shared at runtime but remain encrypted while stored in flash.

For more information on Secure Vault Key Management functionality, see AN1271: Secure Key Storage.
Security Advisories

To subscribe to Security Advisories, log in to the Silicon Labs customer portal, then select Account Home. Click HOME to go to the portal home page and then click the Manage Notifications tile. Make sure that ‘Software/Security Advisory Notices & Product Change Notices (PCNs)’ is checked, and that you are subscribed at minimum for your platform and protocol. Click Save to save any changes.

5.3 Support

Development Kit customers are eligible for training and technical support. Use the Silicon Labs 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.
Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!