Proprietary Flex SDK 3.0.0.2 GA
Gecko SDK Suite 3.0
July 29, 2020

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):

3.0.0.2 released July 29, 2020

Proprietary

CONNECT APPS AND STACK KEY FEATURES

- Studio V5/GSDK 3.0 support for CONNECT
- EFR32xG22 CONNECT GA support
- Improved CONNECT frequency hopping algorithm to handle synchronization loss
- TX and RX packet time stamping to allow accurate synchronization

RAIL LIBRARY KEY FEATURES

- Studio V5/GSDK 3.0 support for RAIL
- New channel hopping (Preamble Sense) and low duty cycle (LDC) modes to help performance in noisy environments
- Optional support for the Silicon Labs Power Manager
- New and updated Sample Applications

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 Applications

1.1 New Items

Added in release 3.0.0.2

- New applications:
  - *Flex (RAIL) – Empty Example*
  - *Flex (RAIL) – Simple TRX with Auto-ACK*
- *RAIL: Duty Cycle* application is split into two applications:
  - *Flex (RAIL) – Burst Duty Cycle*
  - *Flex (RAIL) – Long Preamble Duty Cycle*
- Each *RAIL:Range* application is contained by Flex SDK:
  - *Flex (RAIL) – Range Test*
  - *Flex (RAIL) – Range Test DMP* (was part of Bluetooth SDK)
  - *Flex (RAIL) – Range Test BLE and IEEE802.15.4*
  - *Flex (RAIL) – Range Test BLE and IEEE802.15.4 with DMP* (was part of Bluetooth SDK)
- New easy-to-use application-level components:
  - *Simple RAIL Heartbeat*
  - *Simple RAIL Rx*
  - *Simple RAIL Rx CLI*
  - *Simple RAIL Tx*
  - *Simple RAIL Tx CLI*
- EFR32XG22 support for Connect

1.2 Improvements

Changed in release 3.0.0.2

- Automatic bootloader support for *Flex (RAIL) – Switch* demo application (brd4164a)
- Common Improvements
  - Common code structure.
  - Radio Board i.e. frequency adaptive default Radio Configuration usage (each Radio Configurator-dependent RAIL and Connect application).
- RAIL improvements
  - Each RAIL application has CLI support (except *Flex (RAIL) – WMBUS Meter* and *Flex (RAIL) – WMBUS Collector*).
  - *Flex (RAIL) – Range Test BLE and IEEE802.15.4* and *Flex (RAIL) – Range Test BLE and IEEE802.15.4 with DMP* applications work with all EFR32 families (depending on hardware capabilities) and not just EFR32XG21.
  - *Flex (RAIL) – Simple TRX Multi-PHY* can be used with each dual band board / each subGHz frequency by default.
  - Refactored easy-to-understand statemachine at:
    - Flex (RAIL) – Light.
    - Flex (RAIL) – Switch.
    - *WMBUS applications* support 169/434 MHz.
  - *Flex (RAIL) – Energy Mode* application is compatible (packet format) with *Flex (RAIL) – Simple TRX* application
- Connect improvements
  - *Connect (SoC): Commissioned Device* is renamed to *Flex (Connect) – SoC Direct Mode Device.*

1.3 Fixed Issues

Fixed in release 3.0.0.2

ID #	Description
RAIL_SDK-29	State machine of *Flex (RAIL) - Simple TRX with Auto-ACK* application is refactored – RAIL auto ACK functionality doesn't depend on the PHY

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 https://www.si-labs.com/products/software.

ID #	Description	Workaround
	No Radio Configurator support for EFR32XG21.	
	***Connect NCP-Host applications* are not supported.**	
	***Connect (SoC): Demo Connect Light* and *Connect (SoC): Demo DMP Connect Switch* are not supported**	

1.5 Deprecated Items

None

1.6 Removed Items

Removed in release 3.0.0.2

- RAIL
  - *RAIL: Simple RAIL with HAL* and *RAIL: Simple RAIL without HAL* have been removed. Use (*Flex (RAIL) – Empty Example* instead.
  - *RAIL: Simple TRX with FIFO (Long Packet)* application has been removed.
  - *RAIL: Connected Motion for EFR32 Thunderboard* application has been removed.
- Connect
  - *Connect (SoC): Wire-Replacement* has been removed. Use *Flex (Connect) – SoC Direct Mode Device* instead.

2 Connect Stack

2.1 New Items

None

2.2 Improvements

None

2.3 Fixed Issues

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 https://www.silabs.com/products/software.

ID #	Description	Workaround
	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.	
501561	**In the Legacy HAL component ,the PA configuration is hard-coded regardless of the user or board settings.**	**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.**

2.5 Deprecated Items

None

2.6 Removed Items

None

3 RAIL Library

3.1 New Items

Added in release 3.0.0.2

- Added support for setting the default value of the FramePending bit in outgoing IEEE 802.15.4 ACKs to true. This means that the user would then be responsible for clearing this bit in the frame pending callback instead of having it default to cleared and having to set it. See the RAIL_IEEE802154_Config_t::defaultFramePendingInOutgoingAcks field for configuring this feature.
- Added a new API RAILCb_ConfigSleepTimerSync() to allow for configuration of the PRS and RTCC channels used for timer sync operations.
- Added support for a new RAIL_EVENT_SCHEDULED_RX_STARTED and RAIL_EVENT_SCHEDULED_TX_STARTED, triggered when a scheduled receive or transmit begins. These are the same value because a scheduled receive and transmit cannot occur at the same time. Note: This new event shifted the bit positions of some events in RAIL_Events_t.
- Provided new RAIL_RX_CHANNEL_HOPPING_MODE_MULTI_SENSE along with RAIL_RxChannelHoppingConfigMultiMode_t to configure its parameters. This mode can be configured to tolerate brief loss of timing and/or preamble, making it less susceptible to hopping than the single-sense modes. It can also be used with RX duty cycling.
- Provided new RAIL_RX_CHANNEL_HOPPING_OPTION_RSSI_THRESHOLD to augment each of the RAIL_RxChannelHoppingMode_t modes with one RSSI Threshold check on entering receive for the channel. If the RSSI is below that specified by RAIL_RxChannelHoppingConfigEntry_t::rssiThresholdDbm then hop (or if below RAIL_RxDutyCycleConfig_t::rssiThresholdDbm, suspend Rx). RAIL_RxDutyCycleConfig_t has been augmented not only with this new field, but also now includes RAIL_RxDutyCycleConfig_t::options field. For those options to be recognized by RAIL_ConfigRxDutyCycle() (due to backwards compatibility) RAIL_RxDutyCycleConfig_t::mode must be one of the new WITH_OPTIONS modes, e.g. RAIL_RX_CHANNEL_HOPPING_MODE_TIMEOUT_WITH_OPTIONS.
- Added support for the Silicon Labs Power Manager component. When enabled via RAIL_InitPowerManager(), RAIL will communicate directly with the power manager to configure sleep modes.
- Updated the function RAIL_GetRxPacketDetailsAlt to return the time position of the received packet timestamp corresponding to the default location in the packet.
- Added a new function, RAIL_GetTxPacketDetailsAlt2, which allows a RAIL_TxPacketDetails_t structure to be passed as an argument.
- Added a new function, RAIL_GetTxTimePreambleStartAlt, which allows a RAIL_TxPacketDetails_t structure to be passed as an argument.
- Added a new function, RAIL_GetTxTimeSyncWordEndAlt, which allows a RAIL_TxPacketDetails_t structure to be passed as an argument.
- Added a new function, RAIL_GetTxTimeFrameEndAlt, which allows a RAIL_TxPacketDetails_t structure to be passed as an argument.

3.2 Improvements

Changed in release 3.0.0.2

- Relax constraints in RAIL to allow calling RAIL_SetRxTransitions, RAIL_SetTxTransitions, RAIL_ScheduleRx, and all of the RAIL_BLE_ConfigPhy before the radio is completely IDLE.
- Updated the pa_customer_curve_fits.py helper script to work with Python 3 as well as Python 2.
- Changed pa_customer_curve_fits.py to take maxpower as a parameter to generate better curves. When maxpower and increment are different than the defaults they will now be included in the output curve. The current power curve limits can be read at runtime from the new RAIL_GetTxPowerCurveLimits API.
- The RAIL_GetRadioEntropy() API will now ensure a valid radio configuration has been loaded using RAIL_ConfigChannels() since it can cause problems if the radio is used before this.
- Changed the minimum ramp time of the RAIL_TX_POWER_MODE_2P4GIG_MP to be 3us to avoid problems with shorter ramp times.
- Improved frequency accuracy on EFR32xG1x devices when the radio configuration has an entry with a large number of channels. Previously, small errors in the channel spacing calculation could exist. If they did, then when computing the channel frequency this error would be multiplied by the channel number minus the start channel for that entry causing some drift for higher order channels. This was not much an issue in most cases, but for certain PHY and crystal combinations it could be worse.

3.3 Fixed Issues

Fixed in release 3.0.0.1 Beta 2

ID #	Description
456338	Fixed an issue with RAIL state transitions where an internal timer wrapping could cause incorrect transition times. This error would previously affect a maximum of one packet every 15 minutes.
459581	Fixed an issue where the output power was too low on the EFR32xG21 when using the RAIL_TX_POWER_MODE_2P4GIG_MP PA and certain ramp times.
464534	Fixed issue where RAIL_StartAverageRssi() ran twice as long as it should have.
464734	Regenerated the power curves for the EFR32xG22 to allow access to the maximum power level available on the chip.
464735	Closed tiny timing window on EFR32xG13 that might corrupt PTI appended info when idling the radio.
465096	Fixed an issue where RAIL_Idle() was not properly terminating an ongoing RAIL_StartAverageRssi() process.
466012	Fixed an issue where the CRC could be disabled indefinitely on transmit when switching configs in a multiphy setup.
474678	Fixed an issue with duty cycle receive and channel hopping on the EFR32xG1x parts where some components would be left on even with long delay parameters causing extra current to be used. This allows for a noticeable improvement in power consumption when using the RAIL_ConfigRxDutyCycle() API with a delay in the hundreds of microseconds or more.
475184	Fixed an issue on the EFR32xG22 where the receiver was not automatically re-calibrated if the temperature changed significantly while sitting in receive. This could cause the radio to go off channel for significant temperature changes resulting in receive problems.
477833	Some radio configurations on the EFR32XG22 are not usable with RAIL address filtering and RAIL 802.15.4 filtering. Add an assert to catch those cases.
479539	Fixed a bug where the RAIL_ConfigTxPower() would override the PA capacitor tune values for transmit and receive without caching them. In a multiprotocol scenario this could cause us to apply incorrect PA capacitor tune values set prior to a call to RAIL_ConfigTxPower. Note that you must still call RAIL_SetPaCTune after any making any changes to the power configuration via RAIL_ConfigTxPower.
479665	Fixed an issue where RAIL_SetRxFifo() would not reject a buffer smaller than 64 bytes and would mistakenly think it is very big. In addition, RAIL_ASSERT_FAILED_RX_FIFO_BYTES can no longer occur with a 64 byte buffer when doing IR calibration.
482007	Fixed a bug in multiprotocol RAIL where running an IR calibration during a protocol switch would fail. The calibration function will now return RAIL_STATUS_INVALID_STATE if called in such a scenario.
483688	Fixed an issue where the power mode selected when using RAIL_TX_POWER_MODE_2P4GIG_HIGHEST on supported chips was not saved in a multiprotocol context and could cause problems.
484374	Fixed regression from 2.8.1 on EFR32xG21 where Bluetooth LE did not include packet sync word on PTI.
489214	Fixed an issue where calling RAIL_IEEE802154_CalibrateIr2p4Ghz, RAIL_IEEE802154_CalibrateIrSubGhz, or RAIL_BLE_CalibrateIr with a NULL imageRejection parameter would result in a crash.

3.4 Known Issues in the Current Release

None

3.5 Deprecated Items

None

3.6 Removed Items

None

4 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

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 Launcher Perspective, under this SDK's **Release Notes** doc header.

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*.

4.1 Installation and Use

Stack installation instruction are covered in the Simplicity Studio 5 online User's Guide.

Use the Flex SDK v3.x with the Silicon Labs Simplicity Studio 5 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/.

4.2 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.

5 Legal

5.1 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.

5.2 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-M0+, Cortex-M3, Cortex-M33, Cortex-M4, TrustZone, Keil and Thumb are trademarks or registered trademarks of ARM Holdings.

Zigbee® and the Zigbee logo® are registered trademarks of the Zigbee Alliance.

Bluetooth® and the Bluetooth logo® are registered trademarks of Bluetooth SIG Inc.

All other products or brand names mentioned herein are trademarks of their respective holders.