RSL10 SDK Release Notes - 2.2

October 2, 2018

RSL10 is a Bluetooth® 5 certified multi-protocol radio System-on-Chip (SoC) that brings the industry's lowest power Bluetooth low energy technology to wireless applications. The RSL10 Software Development Kit (SDK) includes the firmware, software, and documentation needed to evaluate this chip and develop applications based on RSL10.

This document lists new features in this release, describes known issues, and points you to more information so you can start using the product.

What's New

Summary	Release Notes
Integrate AES-128 ECB in Remote Microphone applications	AES-128 ECB as an end-to-end encryption, implemented in the Arm Cortex-M3 processor, is now available for the remote microphone (non-coexisting) application.
Setting maximum sleep duration	The default maximum sleep duration has increased to 30 s. In addition, a desired maximum duration time can be set after the `BLE_Initialization()` function by using the `BLE_Sleep_MaxDuration_Set()` function. To reduce the sleep duration, use the function `BLE_Sleep_ReductionTime_set()`. It is zero by default. Both sleep functions accept input arguments in units of 625 us. Do not set the maximum sleep duration to a value where its equivalent number of low power clocks overflows a 32-bit limit.
Adding API to limit Rx window size	Using the `BLE_Set_RxWinSize_Max(uint32_t max_rxWin, uint8_t instant_change_include)` API after `BLE_initialize`, you can set the maximum Rx window size to avoid consuming more power in the case of a poor radio link budget. The default follows the Bluetooth Low Energy standard, so calling this API is usually unnecessary. It is a dynamic API that can be called any time in an application, but it affects all Bluetooth Low Energy links connected to the device. Input arguments are: • max_rxWin: in micro seconds, a zero input means an invalid parameter and it follows default behavior. • instant_change_include: when set to zero, this mechanism is not applied during connection establishment or parameter update procedures; otherwise it is applied in all Bluetooth Low Energy link steps.
Apply Bluetooth Low Energy abstraction to peripheral_server_hrp application	New sample application: ble_peripheral_server_hrp demonstrates how to implement the Heart Rate Sensor portion of the Heart Rate Service, including advertising, pairing, bonding and whitelisting. It is a refactored version of the peripheral_server_hrp sample application.
ble_central_client_scan sample application	New sample application: ble_central_client_scan demonstrates a Bluetooth scanning procedure using the Bluetooth Low Energy abstraction and using the UART CMSIS-driver for input/output.

5 issues

Bug Fixes

Summary	Release Notes
Peripheral Sleep/Standby now sleeping when external clock is used	Fixed issue where the external clock, `RTC_CLK_SRC_DIO`, in *peripheral_server_sleep* and *peripheral_server_standby* was not working for SDK 2.1 or 2.2. Previously, the signal would not go to sleep at all, and would only advertise once. It would also not be possible to connect while the external clock was used.
Negative TX power in Supplemental Calib app	A button press is added in the supplemental_calib and default_manu_info applications to avoid over-writing into the NVR3 memory.
Revised the Bluetooth stack	Improvements: • 2MBPS feature mask issue • Slave MD bit management updated to speed the acknowledgement • Sync window management updated for all procedures (connection, update, connected) • Parameter request procedure updated if peer does not support it

	• Bonding management updated • Updated link loss detection because of slave latency • Test mode initialization and buffer management updated • Priority update function modified • Security manager updated • L2CAP buffer management and fragment updated • Prevent having a software AES request during the data RX which lead to a TX Crypt error • EDCH - Fixed Co-ordinate invalid Curve attack
Receiving Bluetooth Low Energy packets when remote microphone and Bluetooth Low Energy are used together	When a remote microphone protocol event is supposed to be programmed right before a Bluetooth Low Energy event, and the Bluetooth Low Energy event gets preemption, depending on its state and timing, the Bluetooth Low Energy packet might not be received. The required modification has been applied on the remote microphone protocol to address this issue.
Updating connection parameters with new iPhones	iPhones 8 and X that support LLCP parameter updates initiated by slave devices did not accept these parameters from RSL10. Now the stack sets the parameters of this message in a way that iPhone accepts. It is especially important when MFi is used.
Standby application now works at 2 Mbps	The standby application was supporting only 1 Mbps. Now the required changes have been applied on syslib to save and restore the 2 Mbps configuration, allowing the standby application to work at 2 Mbps.
Upgraded the Bluetooth Low Energy core version	Previous SDKs used Bluetooth Low Energy 4.2. Since RSL10 has been certified for Bluetooth Low Energy core specification 5.0, that specification has been applied to SDK 2.2.
Stack no longer crashes when receiving multiple characteristic handle/UUID pairs in a Read By Type Response	When Data Length Extension (DLE) was enabled and a central device discovered characteristics of a peripheral with a large attribute database (at least 25 attributes with a 16-byte UUID), the central device used to crash. This is fixed.
BLE Mesh flash issues	Fixed a BAD_ADDRESS error reported when using the Flash_EraseSector function in the ble_mesh application.
CMSIS-Driver clock restrictions removed	The I2C CMSIS-Driver has been updated to allow unprecise prescale clock values when it's impossible to achieve an exact ratio between the system clock and the desired I2C bus speed.
Further investigation into a crash issue on Mesh	The stack used to try to delete an event that was deleted already, causing a memory crash when Mesh was using advertising and scanning. It has been fixed.
Scrambled Bluetooth address using PRA is no longer the same value after Reset	Updated the *peripheral_server_PRA* sample application to generate a different random address once the device is power cycled.

Removed Sys_ASRC_Enable function	The `Sys_ASRC_Enable` function did not function as intended. Use these methods instead:
	`// Enable ASRC`
	`ASRC_CTRL->ASRC_ENABLE_ALIAS = ASRC_ENABLE_BITBAND;`
	`// Disable ASRC`
	`ASRC_CTRL->ASRC_DISABLE_ALIAS = ASRC_DISABLE_BITBAND;`

13 issues

Migrating from RSL10 SDK 2.1 to 2.2

If you use Release 2.1 code or projects in Release 2.2, remember the following:

- Compatibility of the Remote Microphone protocol is not guaranteed between SDK 2.1 and SDK 2.2. In other words, an application compiled as TX in SDK 2.1 will not necessarily work with an RX application compiled in SDK 2.2.
- Although the Release 2.1 sample code builds successfully in Release 2.2, backwards compatibility is not guaranteed.

Known Issues

The following issues are known to be outstanding in release 2.2:

Summary	Release Notes
Sleep App: Connection drop when using RC_OSC and an external clock	A bug exists with the *peripheral_server_sleep* app, in which the connection drops between any central server and an RSL10 board if an external clock or the `RTC_CLK_SRC_RC_OSC` is used with `LOW_POWER_CLK_UPDATE` disabled. As a temporary fix to avoid a dropped connection, enable `LOW_POWER_CLK_UPDATE` in *app.h*.
Sync_pulse DIO initial for remote_mic_coded app	Initialize `DIO_SYNC_PULSE` as a `GPIO_OUT` in the *remote_mic_trx_coded* app. `Sys_DIO_Config(DIO_SYNC_PULSE, DIO_MODE_GPIO_OUT_0);`
I2C CMSIS Driver "xfer_pending" not supported	The I2C CMSIS-Driver does not support the "xfer_pending" feature due to a hardware limitation. Therefore, generating a repeated start condition on the I2C interface using the I2C CMSIS-Driver is not possible.
Remote Mic TX with Audio Shield does not work after Supplemental Calib app	To make the *remote_mic_tx_raw* application to work with the Audio Shield, erase the NVR3 program that is loaded by the supplemental calibrate application.
CRC limitation with 89 vector table entries	The CRC function is not valid if all 88 possible vectors are in use.
Controller privacy cannot be used with some smartphones	Because of a hardware issue, smartphones Samsung S7 and S8 do not show RSL10 advertisements when controller privacy is used. Therefore, use host privacy in these cases. When host privacy is used, you can decrease power consumption close to the level of controller privacy use by employing a new feature called "power save control". To enable it, see the new API called `GAPM_POWER_SAVE_CTRL_CMD` as specified in the GAP Interface Specification from CEVA.
XTAL 48MHz start up time needs to be increased	When the RC oscillator is used as the source of the low power clock, the Bluetooth low energy connection with the sleep application is not very stable and sometimes a disconnection is observed. To alleviate the issue and also account for board to board variation, the wake-up delay for this case has been increased from 1.1 ms to 1.3 ms by calling `BLE_LLD_Sleep_Params_Set` with the right parameter.

Delay needed in the remote_mic_rx_raw sample application for PCM input	To have good audio quality for PCM input, add a delay in the `ASRC_out` DMA interrupt. Use a delay instruction in the `DMA_IRQ_FUNC(ASRC_OUT_IDX)(void)` function in *app_func.c* as shown below:
	`#if (INPUT_INTRF == PCM_RX_RAW_INPUT)`
	`  Sys_Delay_ProgramROM(100);`
	`#endif`

Remote Mic TX: smaller sub-frame causes noisy right channel	Choosing a SUBFRAME that is less than 32 samples for the DMIC and PCM ports in the Stereo Raw Audio Stream Broadcast Transmitter Custom Protocol sample application (*remote_mic_tx_raw*) causes periodic noise on the right channel for PCM and both channels for DMIC.
Investigating increase in latency for custom_protocol_trx	Configuring the RX output to SPI causes an extra 1.5 ms increase in the end-to-end latency. Now it is 8.6 ms, but it is supposed to be 7 ms.
Investigating spurious emissions for the remote mic protocol	The remote microphone protocol implementation on the transmitter side has spurious emission violation non-conformance with regulatory tests.
Installer fails if the path is too long	If the installation path is longer than 50 characters, the installer fails with an error specifying which path is too long. When the installer tries to install some of the subfolders, the entire path length exceeds the maximum allowed by Windows. If you choose a location that is different from the default installation location, ensure that it does not exceed 50 characters.
Remote microphone performance needs further evaluation	The audio stream broadcast custom protocol has not been fully characterized.

13 issues

Further Reading

See *Getting Started with RSL10* to discover:

- How to install the software development tools
- How to connect to the Evaluation and Development Board
- How to run a simple sample application that shows the board is working
- An overview of the documentation set

Release Organization

The following files are available on the same web page as these release notes:

Release Item	Description
RSL10_getting_started.pdf	Second document you should read after this one. For a full description, see Further Reading.
RSL10_SDK_Installer.msi	Installer for the standard RSL10 SDK (Oxygen Eclipse)
ONSemiconductor.RSL10.2.2.0.pack	Arm CMSIS-Pack for Oxygen
Documentation.zip	All manuals in one folder that can be accessed without installing or starting an IDE
RSL10_Utility_Apps.zip	Contains utility applications and external projects that can be used to help demonstrate and test the functionality of the RSL10 Evaluation and Development Board: - Standalone Flash Loader - Ezairo 7100-based sample audio applications
RSL10_LPDSP32_Support.zip	Startup files, Eclipse project files, a utility to integrate LPDSP32 output code with Arm Cortex®-M3 processor code, a test suite, and documentation
RSL10_release_note_history.zip	An archive file containing previous versions' release notes.

Copyright © 2018 Semiconductor Components Industries, LLC (d/b/a ON Semiconductor).